Dreamweaver +ASP.NET 动态网站设计与典型实例

彭为 陶利 陈晓明 王鹏 编著

- 由多名动态网站开发资深专家合力编写，内容实用
- 从开发环境设置到动态网站开发全程实例逐步演示
- 含大量完整的ASP.NET动态网站版块开发典型实例
- Dreamweaver、ASP.NET、数据库技术综合开发应用

清华大学出版社

Dreamweaver+ASP.NET 动态网站设计与典型实例

彭　为　陶　利　陈晓明　王　鹏　编著

清华大学出版社

北　京

内 容 简 介

Dreamweaver 8 是 Macromedia Studio 8 Web 开发套件的一个重要组成部分，它在原有版本的基础上，新增了许多激动人心的功能。ASP.NET 是 Microsoft 推出的新一代动态 Web 编程技术，该技术的推出是 Web 应用开发领域的一次革命性突破。

本书围绕 Dreamweaver 8 的使用，由浅入深，从基础到应用，着重叙述如何在 Dreamweaver 8 中开发基于 ASP.NET 的 Web 应用程序，为读者对 Dreamweaver 8 和 ASP.NET 的结合使用进行了全面的深入讲解。内容涉及 Dreamweaver 8 使用、ASP.NET 语法基础、Web 表单和控件、验证控件、ADO.NET 数据访问、动态网站各个版块的制作过程等。

本书结构清晰，内容丰富，实例详尽，适合于使用 Dreamweaver 8 开发 ASP.NET Web 应用的各层次的用户，也可以作为广大院校和培训机构相关专业学生的培训教材。

本书封面贴有清华大学出版社防伪标签，无标签者不得销售。
版权所有，侵权必究。侵权举报电话：010-62782989　13501256678　13801310933

图书在版编目(CIP)数据

Dreamweaver+ASP.NET 动态网站设计与典型实例/彭为，陶利，陈晓明，王鹏编著. —北京：清华大学出版社，2008.1
ISBN 978-7-302-16549-1

Ⅰ. D…　Ⅱ. ①彭…　②陶…　③陈…　④王…　Ⅲ. ①主页制作—图形软件，Dreamweaver　②主页制作—程序设计　Ⅳ. TP393.092

中国版本图书馆 CIP 数据核字(2007)第 184263 号

责任编辑： 黄　飞
封面设计： 杨玉兰
版式设计： 北京东方人华科技有限公司
责任校对： 李玉萍
责任印制：

出版发行： 清华大学出版社　　**地　　址：** 北京清华大学学研大厦 A 座
http://www.tup.com.cn　　**邮　　编：** 100084
c-service@tup.tsinghua.edu.cn
社 总 机： 010-62770175　　**邮购热线：** 010-62786544
投稿咨询： 010-62772015　　**客户服务：** 010-62776969
印 刷 者：
装 订 者：
经　　销： 全国新华书店
开　　本： 185×260　**印　张：** 27.5　**字　数：** 661 千字
附光盘 1 张
版　　次： 2008 年 1 月第 1 版　　**印　次：** 2008 年 1 月第 1 次印刷
印　　数： 1~4000
定　　价： 46.00 元

本书如存在文字不清、漏印、缺页、倒页、脱页等印装质量问题，请与清华大学出版社出版部联系调换。联系电话：010-62770177 转 3103　　产品编号：

前　　言

ASP.NET 是微软最新推出的新一代 Web 开发技术，它也是微软所推出的新型体系结构.NET 的一个重要组成部分。在针对 ASP.NET 的开发工具中，Dreamweaver 8 的出现无疑使之成为 ASP.NET 开发人员的最佳选择。

Dreamweaver 8 是 Macromedia 公司推出的一个专业可视化的 HTML 编辑工具，用于对 Web 站点、Web 页面和 Web 应用程序进行设计、编码和开发。Dreamweaver 8 也是 Macromedia 公司所推出的 Macromedia Studio 8 Web 开发套件的一个重要组成部分。Dreamweaver 8 在原有版本的基础上，新增了许多激动人心的新功能，进一步改善了 Dreamweaver 的易用性和扩展性，使用户无论处于设计环境还是编码环境都可以方便、快捷地创建 Web 页面。

本书从 Dreamweaver 8、ASP.NET 的基础知识入手，结合基本的开发知识、技巧和典型的开发实例，循序渐进地介绍了 Dreamweaver 8 和 ASP.NET 环境下各种动态网站模块的开发。本书共分 11 章，内容涉及 Dreamweaver 8 的使用、ASP.NET 语法基础、Web 表单和控件、验证控件、ADO.NET 数据访问、模板控件等，各章内容介绍如下。

第 1 章　Dreamweaver 8 基础知识：介绍 Dreamweaver 8 基础知识，包括 Dreamweaver 8 中的新增功能、软件安装及常用的操作界面。

第 2 章　站点操纵：介绍在 Dreamweaver 8 中对站点的相关操作，包括新建站点、管理站点及一些高级操作等。

第 3 章　Dreamweaver 8 的基本操作：介绍在 Dreamweaver 8 中如何操作相关的基本元素，包括文本、图像、链接以及表格等。

第 4 章　数据库的操作：介绍 Access 数据库的基本操作。

第 5 章　VB.NET 语言基础及 ASP.NET 网页基本结构：介绍 VB.NET 语言基本语法以及 ASP.NET 页面基本结构，包括 ASP.NET 页面中的代码块、表单和常用的服务器控件。

第 6 章　ADO.NET 与数据库的访问：讲解如何在 ASP.NET 中使用 ADO.NET 进行数据访问，并通过实例介绍 DataGrid、DataList 和 Repeater 等常用数据控件的使用。

第 7 章　留言簿制作：介绍如何在 Dreamweaver 8 中设计制作基于 ASP.NET 的留言簿。

第 8 章　会员管理：介绍如何实现网站中常用的基于 ASP.NET 的会员管理系统。

第 9 章　网络相册：介绍如何在 Dreamweaver 8 中设计制作基于 ASP.NET 的网络相册。

第 10 章　同学录：介绍如何在 Dreamweaver 8 中设计制作基于 ASP.NET 的同学录。

第 11 章　动态新闻发布系统：介绍如何在 Dreamweaver 8 中设计制作基于 ASP.NET

的动态新闻发布系统。

本书重在结合作者的开发经验，充分考虑到读者在学习和应用开发过程中容易出现的错误，避免读者在学习过程中走弯路。全书语言简洁，实例丰富，介绍的技术先进，知识面广，实例典型，十分适合即将走向工作岗位的各级各类学校的学生使用。

本书由彭为、陶利、陈晓明、王鹏、李水、胡勇辉、杭志、兰湘涛、胡容、杭志、文龙等编写。因时间仓促，水平有限，尚有许多不是和不准确之处，恳请广大读者提出批评和建议，也欢迎广大读者到 www.54pub.com < http://www.54pub.com>来进行广泛的交流与探讨。

编　者

目　　录

第 1 章　Dreamweaver 8 基础知识

Dreamweaver 是 Macromedia 公司(已被 ADOBE 公司收购)推出的一个专业可视化的 HTML 编辑工具，用于对 Web 站点、Web 页和 Web 应用程序进行设计、编码和开发。Dreamweaver 的功能强大且界面友好，使用它可制作跨平台浏览的网页。Dreamweaver 所提供的可视化编辑功能，可帮助用户快速创建不需要手工编写任何代码的动态页面。此外，Dreamweaver 还提供了功能全面的编码环境，其中包括代码编辑工具，以及有关层叠样式表(CSS)、JavaScript 和 CodeFusion 标记语言(CFML)等方面的参考资料。

本章将介绍 Dreamweaver 8 的基础知识，以及如何安装 Dreamweaver 8，让读者对 Dreamweaver 8 有一个基本的认识。

1.1　Dreamweaver 8 的新增功能

Dreamweaver 8 是 Macromedia 公司推出的 Macromedia Studio 8 Web 开发套件的一个重要组成部分。Dreamweaver 8 在原有版本的基础上，新增了许多激动人心的新功能，进一步改善了 Dreamweaver 的易用性和扩展性，使用户无论处于哪个开发环境 (即是在设计环境还是编码环境)，都可以方便地创建页面。下面，就来认识一下 Dreamweaver 8 的新增功能或增强功能。

1. 改进的工作环境

Dreamweaver 8 在工作环境方面的改进主要体现在工作区布局的改进、代码折叠的实现以及新增的编码工具栏和样式呈现工具栏。

1)　工作区布局

Dreamweaver 8 提供了两种工作区布局样式：设计器和编码器。

设计器是一个使用 MDI(多文档界面)的集成工作区，其中全部文档窗口和面板被集成在一个较大的应用程序窗口中，同时面板组停靠在右侧，如图 1.1 所示。

编码器同样是一个集成的工作区，但其面板组停靠在左侧，且文档窗口在默认情况下显示为【代码】视图，如图 1.2 所示。

首次启动 Dreamweaver 8 时，系统将会弹出一个对话框让用户选择一种默认的工作区布局。而在以后的操作中，用户可随时切换工作区。

除了以上两种工作区布局样式外，用户还可以根据自己的需要自定义工作区的布局样式，将其保存后，Dreamweaver 8 将记住指定布局中的面板以及其他属性，如面板的位置和大小、面板的展开或折叠状态以及应用程序窗口的位置和大小等。

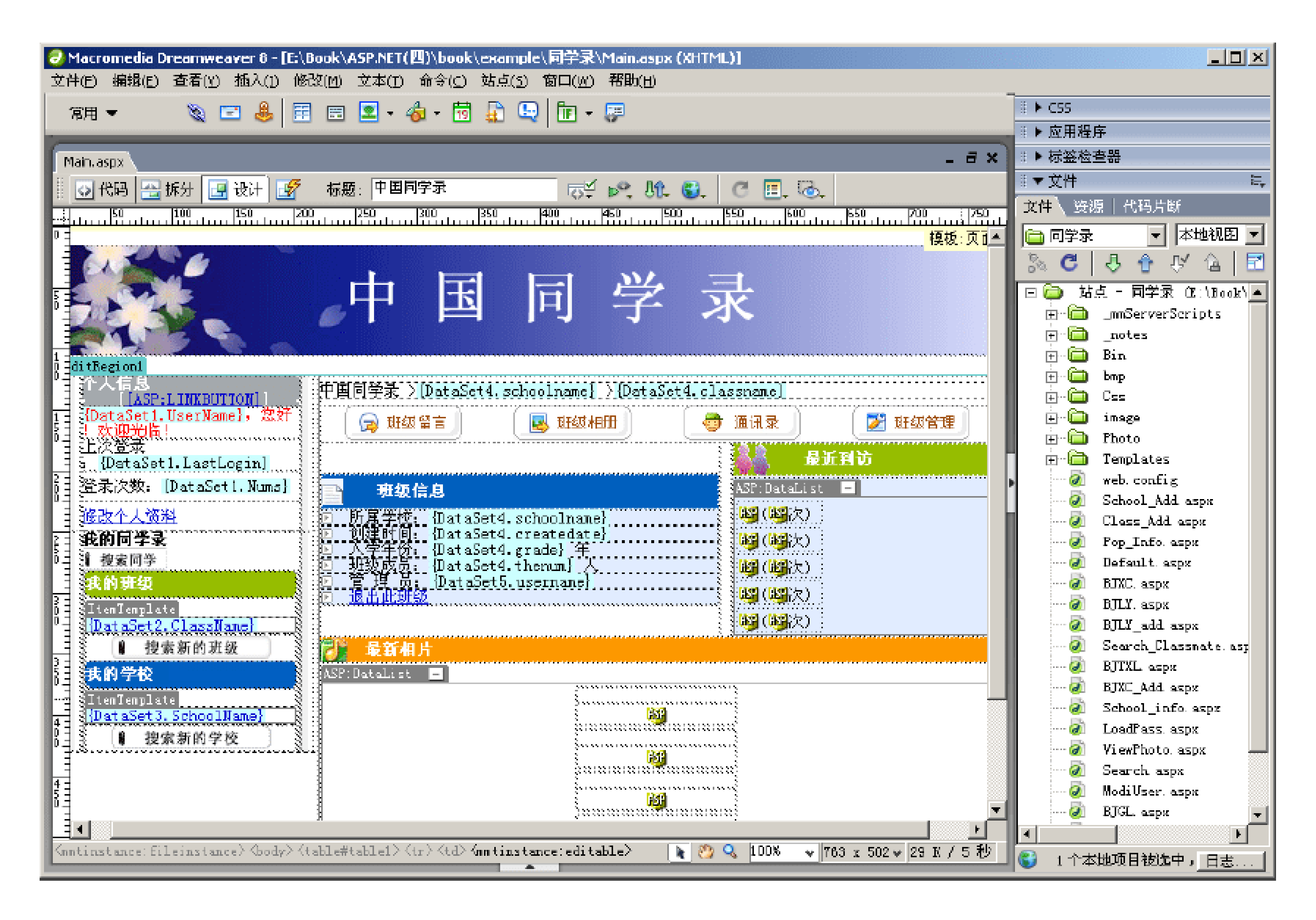


图 1.1　工作区布局样式——设计器

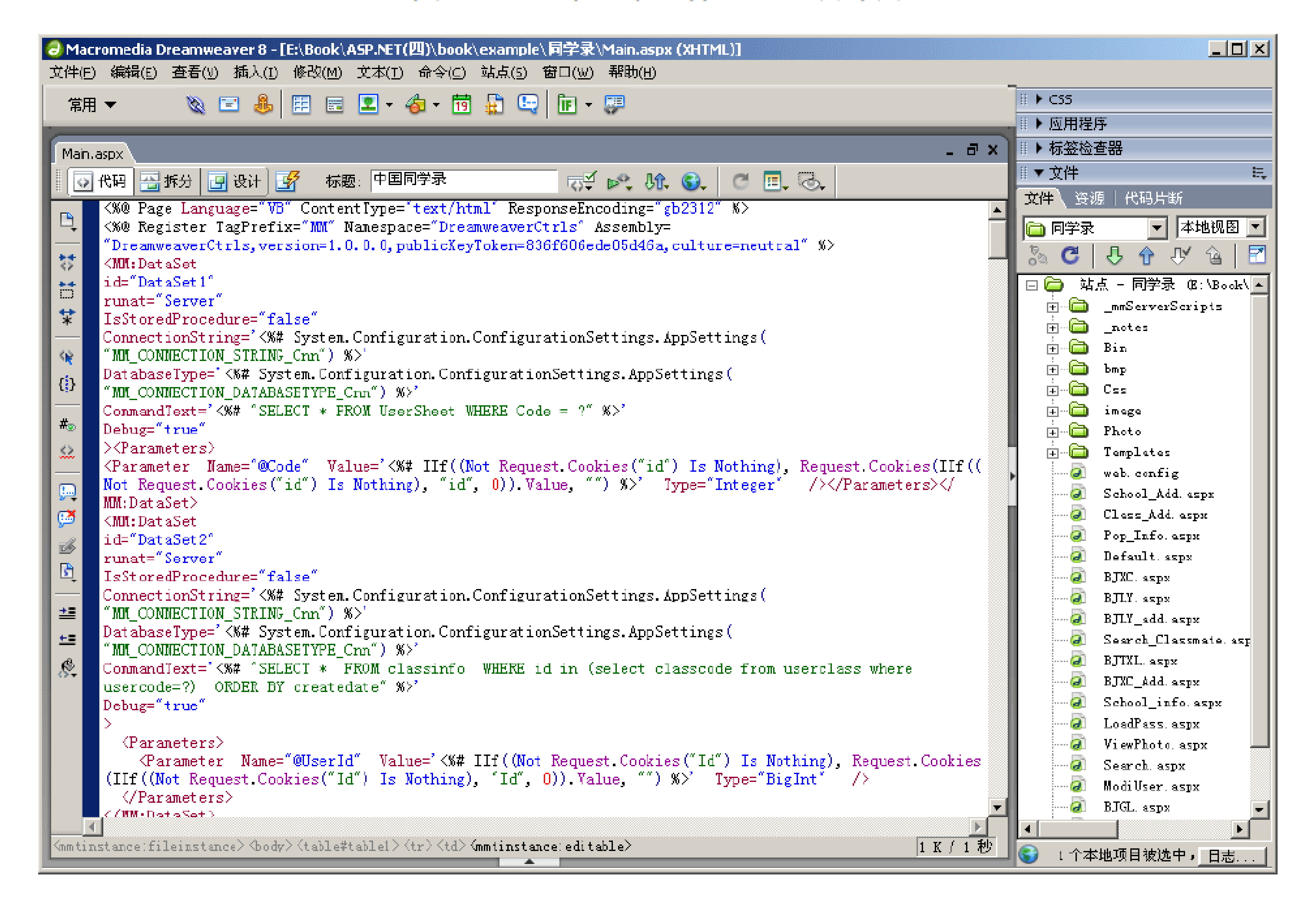


图 1.2　工作区布局样式——编码器

2)　代码折叠

Dreamweaver 8 提供了代码折叠的功能，即将指定的开始标签和结束标签之间的代码块或用户所选定的代码块折叠起来，达到隐藏的目的，以便重点显示用户所要查看的代码。被折叠的代码片断仅显示隐藏的第一行的每一个单词和省略号(…)，同时背景为灰色，带圆角矩形框，如图 1.3 所示。

从以上折叠代码所在的行数(第 101 行)和下一行代码所显示的行数(第 171 行)可以看出，

所隐藏的代码片断包括 70 行(从 101 行到 170 行)代码。当将鼠标放在折叠代码片断的上时，可查看折叠代码片断中的代码而不需扩展该代码片断；双击折叠代码片断，可展开显示代码片断中的所有代码。

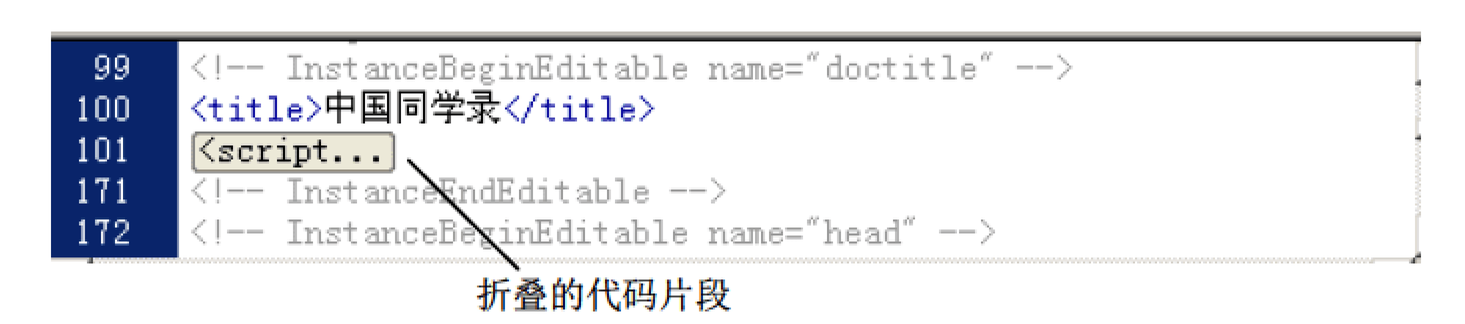


图 1.3 代码折叠

3) 【编码】工具栏

Dreamweaver 8 提供了一个新的工具栏——【编码】工具栏，这是 Dreamweaver 8 以前的版本中所没有的。【编码】工具栏一般停靠在【代码】视图的左侧，它提供了用于常见编码功能的相关按钮，如图 1.4 所示。

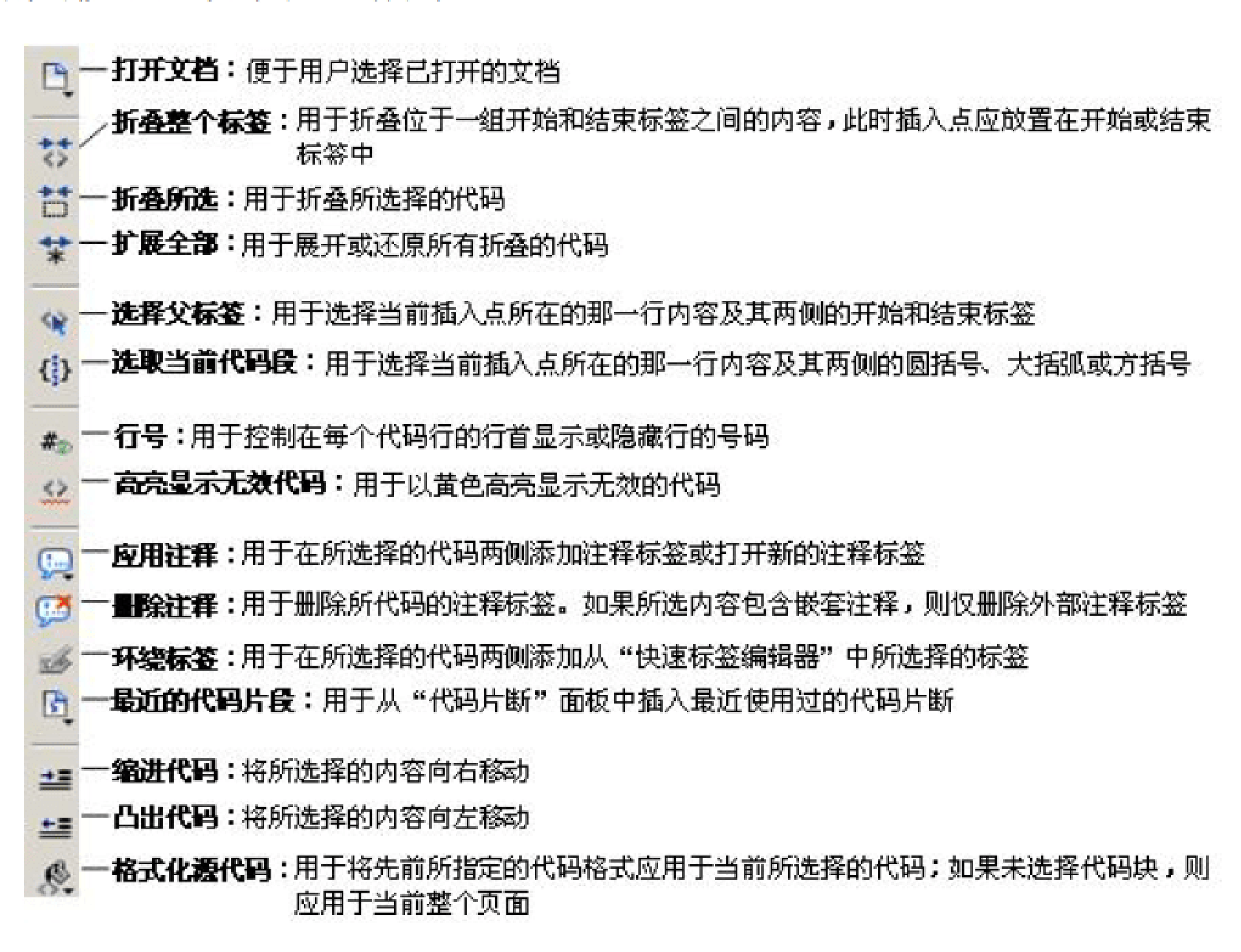


图 1.4 【编码】工具栏

4) 【样式呈现】工具栏

【样式呈现】工具栏是 Dreamweaver 8 提供的一个新特性，它为用户提供了一个快捷的方式来为不同的媒体类型，如屏幕、手持设备和打印输出等进行设计。如图 1.5 所示，【样式呈现】工具栏中包含了一些按钮，这些按钮能够查看设计在不同媒体类型中的呈现方式。

图 1.5 【样式呈现】工具栏

- 屏幕媒体　用于显示页面在计算机屏幕上的显示方式。
- 打印媒体　用于显示页面在打印纸张上的显示方式。
- 手持型媒体　用于显示页面在手持设备(如手机)上的显示方式。
- 投影媒体　用于显示页面在投影设备上的显示方式。
- TTY 媒体　用于显示页面在电传打字机上的显示方式。
- TV 媒体　用于显示页面在电视屏幕上的显示方式。
- 切换 CSS 样式显示　用于控制是否启用 CSS 样式。

提示：只有当文档使用了依赖于媒体的样式表时，【样式呈现】工具栏才有用。所谓依赖于媒体的样式表，是指通过样式表为相应的媒体类型指定某种正文规则。

通过【样式呈现】工具栏，只需创建相应的 CSS 文件并将其应用到页面样式，然后将其和相应的媒体类型匹配，即可快速完成针对不同媒体类型的设计工作。

2. 新增的设计工具

在设计工具方面，Dreamweaver 8 新增了【缩放】工具、辅助线、【插入 Flash 视频】对话框等。

1)　【缩放】工具

Dreamweaver 8 可以通过【缩放】工具放大或缩小当前文档，以便更容易地查看图形的像素精确度、对齐图像、选择较小的图像以及查看较小的文本等。

【缩放】工具一般显示在文档窗口的右下角，它包括一个放大镜图标和一个用于设置缩放比率的文本框，如图 1.6 所示。

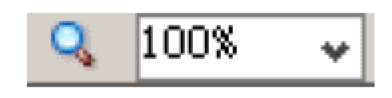


图 1.6　【缩放】工具

在用于设置缩放比率的文本框中，除提供了可以设置缩放的比率，还提供了【符合所选】、【符合全部】和【符合宽度】3 个选项。其中，【符合所选】选项可使所选择的内容填充文档窗口；【符合全部】选项可使整个页面填充文档窗口；【符合宽度】选项可使页面的整个宽度填充文档窗口。

提示：【缩放】工具仅在【设计】视图中可用。

2)　辅助线

辅助线是拖动标尺到文档上时所产生的线条，它有助于用户更加准确地放置和对齐文档上的各种对象，同时也可被用来获取页面元素的大小。在图 1.7 中的虚线即为 Dreamweaver 8 提供的辅助线。

辅助线包括水平辅助线和垂直辅助线两种，其创建方法很简单，只需从相应的标尺用鼠标进行拖动即可。对于已经创建的辅助线，可通过再次拖动来进行重新定位。如果需要查看辅助线分割出来的方格的高度和宽度，可将鼠标移至该方格中，按下 Ctrl 键。在图 1.7 中，两个方块中显示的即为该方格的高度和宽度。

此外，可对辅助线进行相应的操作和设置，如锁定辅助线、将辅助线靠齐元素、将元素靠齐辅助线、改变辅助线的颜色等。还可通过专门的菜单命令对辅助线进行操作，如图 1.8 所示。

3)　【插入 Flash 视频】对话框

Dreamweaver 8 提供了直接在页面中插入 Flash 视频文件的功能，即通过如图 1.9 所示

的【插入 Flash 视频】对话框来快捷地插入 Flash 视频。

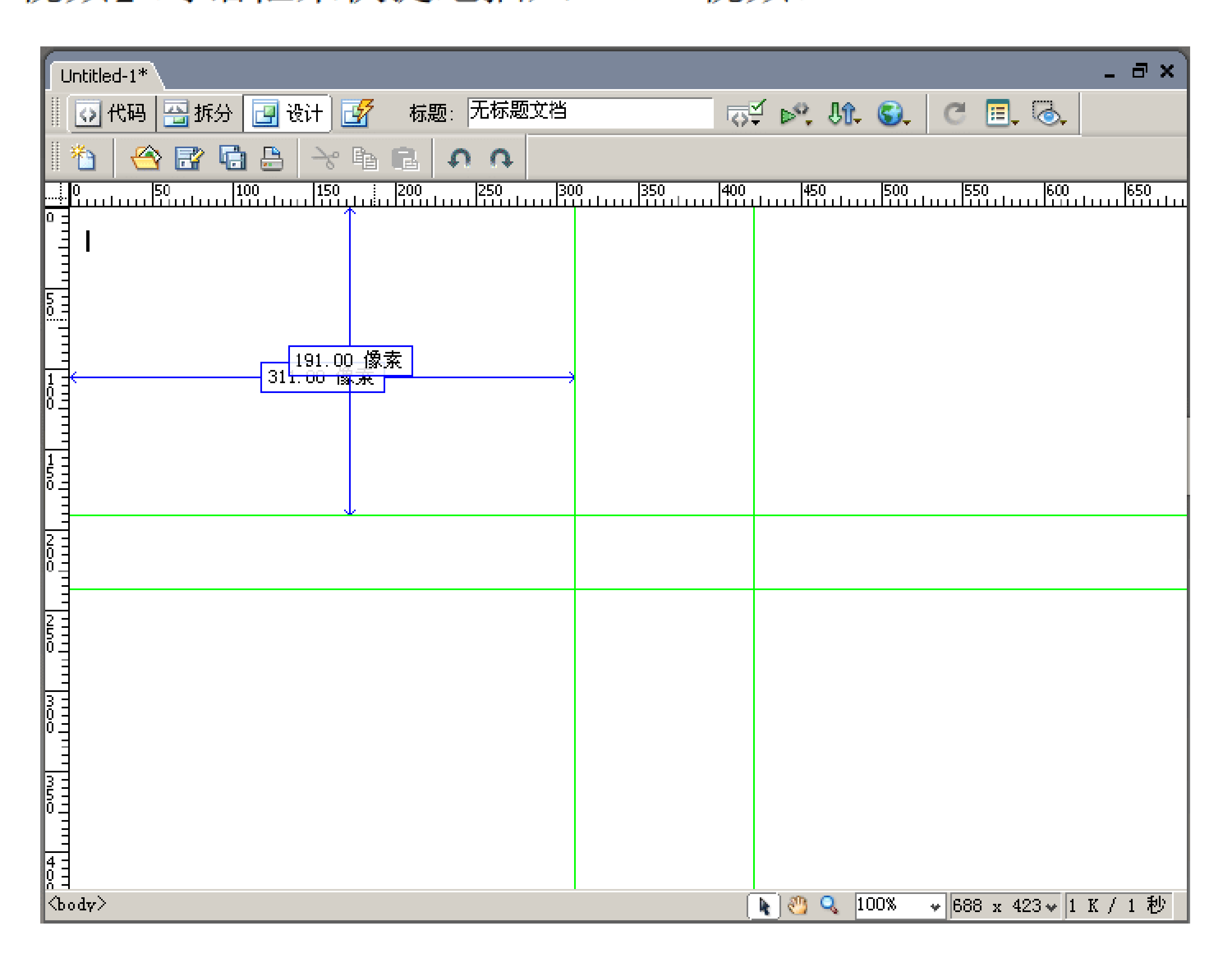


图 1.7　辅助线

显示辅助线(S)　Ctrl+;
锁定辅助线(L)　Ctrl+Alt+;
靠齐辅助线(N)　Ctrl+Shift+;
辅助线靠齐元素(E)　Ctrl+Shift+/
编辑辅助线(E)...
清除辅助线(C)
640 x 480，默认
640 x 480，最大化
800 x 600，最大化
832 x 624，最大化
1024 x 768，最大化
WebTV

图 1.8　辅助线菜单

图 1.9　【插入 Flash 视频】对话框

在【视频类型】下拉列表框中，包括【累进式下载视频】和【流视频】两个选项。其中，累进式下载视频(Progressive Download Video)是将 Flash 视频文件下载到站点访问者的硬盘上再进行播放；而流视频(Streaming Video)则是将 Flash 视频文件进行流处理并立即在 Web 页面中进行播放。

在【外观】下拉列表框中，提供了包含 Flash 视频内容的 Flash 视频组件的外观选项。

通过【宽度】和【高度】两个文本框，可设置以像素为单位的 Flash 视频文件的显示宽度和高度。

在 Web 页面中插入 Flash 视频文件后，可以通过【属性】面板继续修改视频文件的相关属性，如图 1.10 所示。

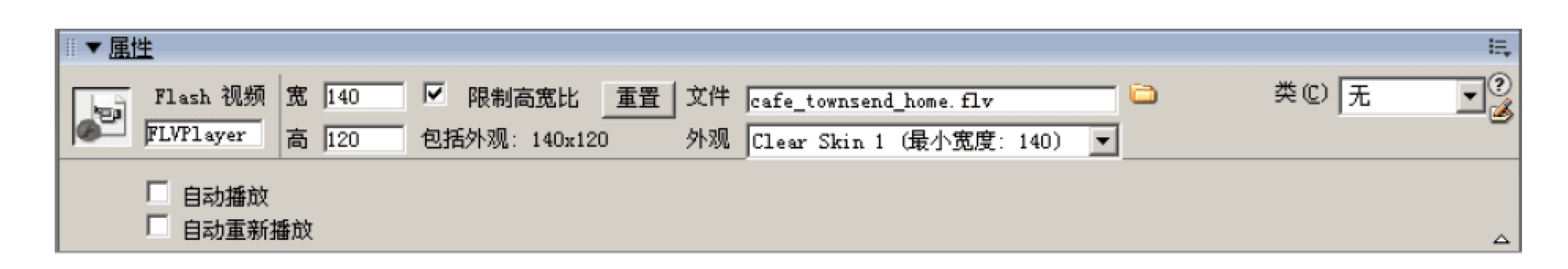


图 1.10　用来设置 Flash 视频文件属性的【属性】面板

3. 扩展的 CSS 功能

Dreamweaver MX 2004 已经具备了功能强大的 CSS 功能。在此基础上，Dreamweaver 8 又提供了更强的支持。

4. 统一的 CSS 面板

Dreamweaver 8 对 CSS 面板进行了重新设计，将 Dreamweaver MX 2004 中的多个 CSS 面板集成到了一个统一的面板中，如图 1.11 所示。

在【全部】模式下，可以查看和设置当前页面中的所有 CSS 样式；在【正在】模式下，可以查看和设置当前所选择的页面元素的 CSS 样式。

在 CSS 面板中提供了 3 种视图：【类别】视图、【列表】视图和【设置属性】视图。其中，【类别】视图将 Dreamweaver 所支持的 CSS 属性划分为 9 个类别：字体、背景、区块、边框、方框、列表、定位、扩展和表，用户可通过该视图查看相应类别的 CSS 属性；【列表】视图则按字母顺序显示 Dreamweaver 所支持的所有 CSS 属性；【设置属性】视图则仅用来显示那些已经设置的 CSS 属性。

此外，在 CSS 面板的右下角还提供了 4 个操作按钮：(附加样式表)、(新建 CSS 规则)、(编辑样式)和(删除 CSS 规则)。

事实上，CSS 面板已经变成了一个更具有可操作性的控制面板，通过它可以快捷地确认样式、编辑样式和查看应用于页面元素的样式。

5. CSS 布局的可视化

在 Dreamweaver 8 中，当工作窗口为【设计】视图时，允许显示 CSS 布局块。所谓 CSS 布局块，是一个 HTML 页面元素，它可以定位在页面中的任何位置。更确切地说，CSS 布局块是 display 属性的属性值不为 inline 的 div 标签，或者是包括 display:block、position:absolute 或 position:relative 等 CSS 声明的任何其他页面元素。

出于可视化呈现的目的，CSS 布局块不包含内联元素(即代码为一行的元素)或段落之类的简单块元素。

Dreamweaver 8 提供了多个命令帮助用户查看 CSS 布局块，如图 1.12 所示。

【CSS 布局背景】命令可显示各个 CSS 布局块的临时指定背景颜色，并隐藏通常出现在页面上的所有背景颜色或图片；【CSS 布局框模型】命令可显示所选择的 CSS 布局块的框模型(即填充和边界)；【CSS 布局外框】命令则可显示页面上所有 CSS 布局块的外框。

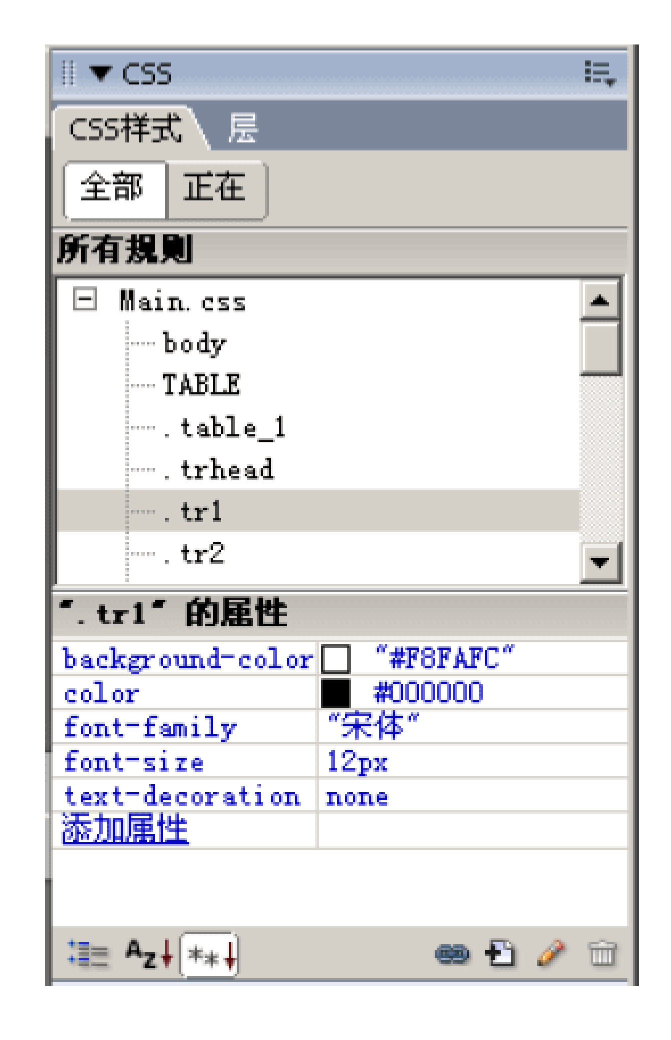


图 1.11　CSS 面板

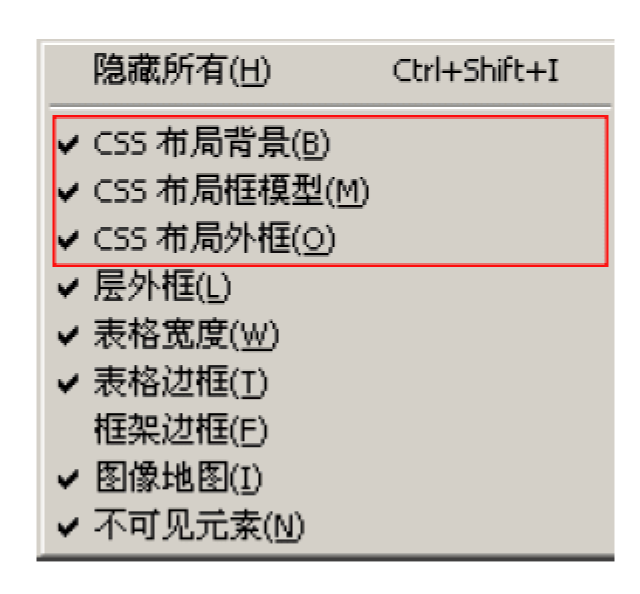


图 1.12　用于查看 CSS 布局块的命令

6. 支持 XML/XSLT 数据的创建

Dreamweaver 8 新增的 XML/XSLT 创建功能简化了用于 Web 浏览的 XML 文件的格式化过程。用户可以创建 XSLT 文件，并完全可以使用 CSS 格式，将其转换成难于理解的 XML 文件，并放入网页中。

Dreamweaver 8 还提供了简捷的方法帮助用户快速格式化一个 XML 文件，以便直接在浏览器中进行浏览。同时，也可使用 Dreamweaver 中的动态页面开发工具，利用 ASP、PHP、ASP.NET 等 Dreamweaver 8 所支持的动态页面开发技术，将 XML 文件转换为任何 Web 浏览器都支持的 HTML 文件。

7. 对文件的处理

1)　文件差异的比较

Dreamweaver 8 提供了一种文件比较工具(即 Diff 工具)。使用该工具可以比较同一文件的本地版本和远程版本的代码、两个不同的远端文件的代码或两个不同的本地文件的代码。这为比较两个文件的差异、进行文件的版本控制提供了极大的便利。

提示： 文件比较工具不是 Dreamweaver 8 自带的，使用前必须进行下载和安装。同时，还必须在 Dreamweaver 8 中指定该工具。

2)　文件的后台传输功能

在 Dreamweaver 8 以前的版本中，当向 Web 服务器进行文件传输时，用户必须在文件传输完成后，才能进行其他工作。而 Dreamweaver 8 提供了 FTP 的后台传输功能，这使得在文件传输时可进行其他操作，如新建和编辑页面等。

此外，Dreamweaver 8 还提供了一种功能强大的文件同步功能，这可以确保 Web 服务器上的内容始终是最新的。

1.2　Dreamweaver 8 的安装

通过上一节的学习，应对 Dreamweaver 8 有一个基本的了解。在这一节中，将向大家介绍 Dreamweaver 8 运行时所必须具备的系统环境及具体的安装过程。

1.2.1　系统要求

Dreamweaver 8 对系统的运行环境要求如下。

- CPU：800 MHz Intel Pentium Ⅲ处理器(或相当等级)或更高
- 内存：256 MB RAM 或更高
- 磁盘空间：至少 650 MB 可用磁盘空间
- 操作系统：Windows 2000/ XP

1.2.2　Dreamweaver 8 的安装过程

Dreamweaver 8 安装文件的大小约为 60MB。下面，具体介绍 Dreamweaver 8 的安装过程。

(1) 双击安装文件，如果安装文件是一个压缩文件，系统将首先将其解压。如图 1.13 所示。

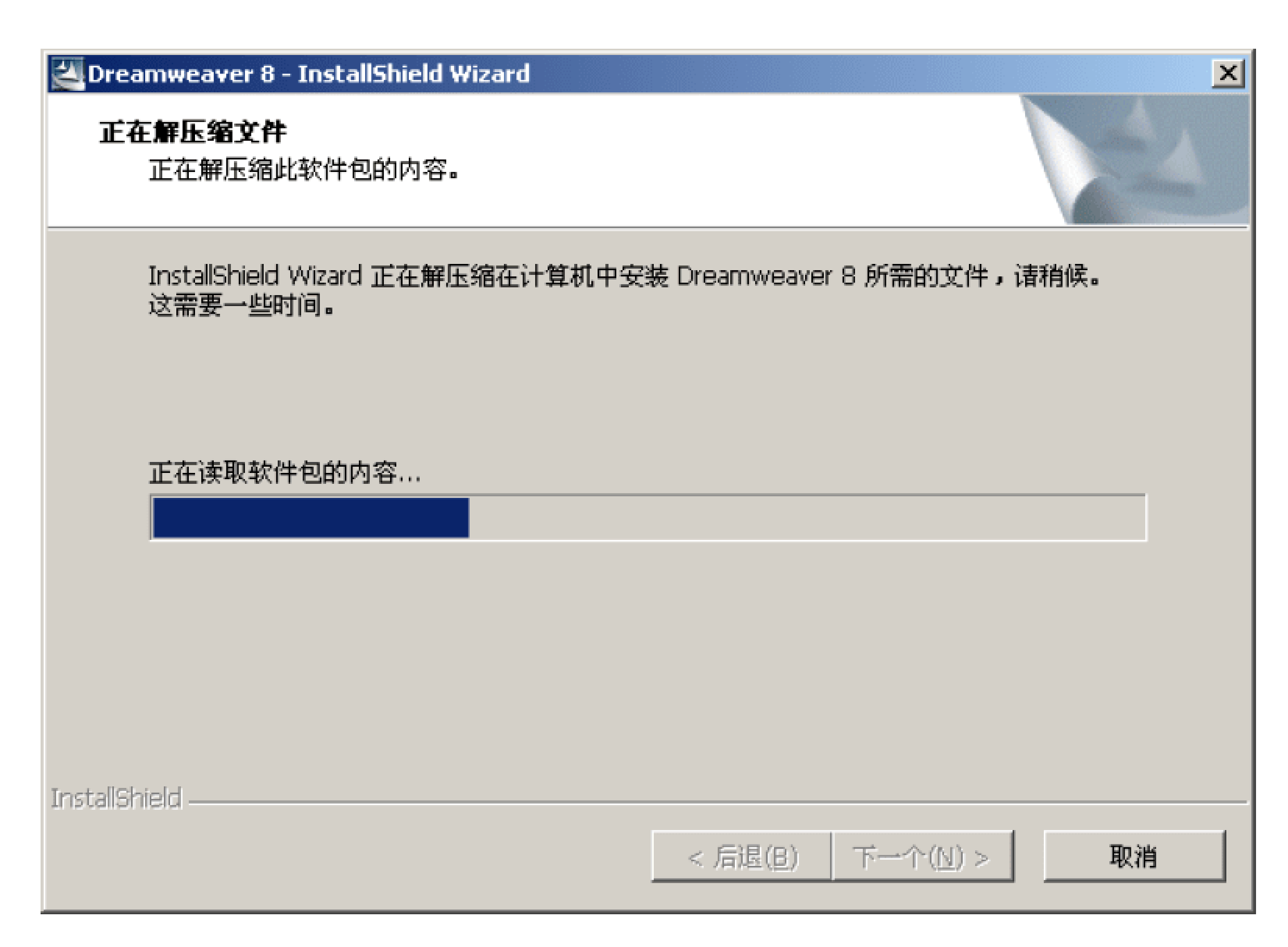


图 1.13　解压文件

(2) 解压文件完毕后，系统将弹出一对话框，提示系统正在准备安装，即进行安装前的准备工作，如图 1.14 所示。

(3) 准备工作完成后，系统将打开安装向导的第一个对话框，如图 1.15 所示。

图 1.14　正在准备安装

图 1.15　【欢迎使用】对话框

(4)　单击【下一步】按钮，进入【许可证协议】对话框，如图 1.16 所示。

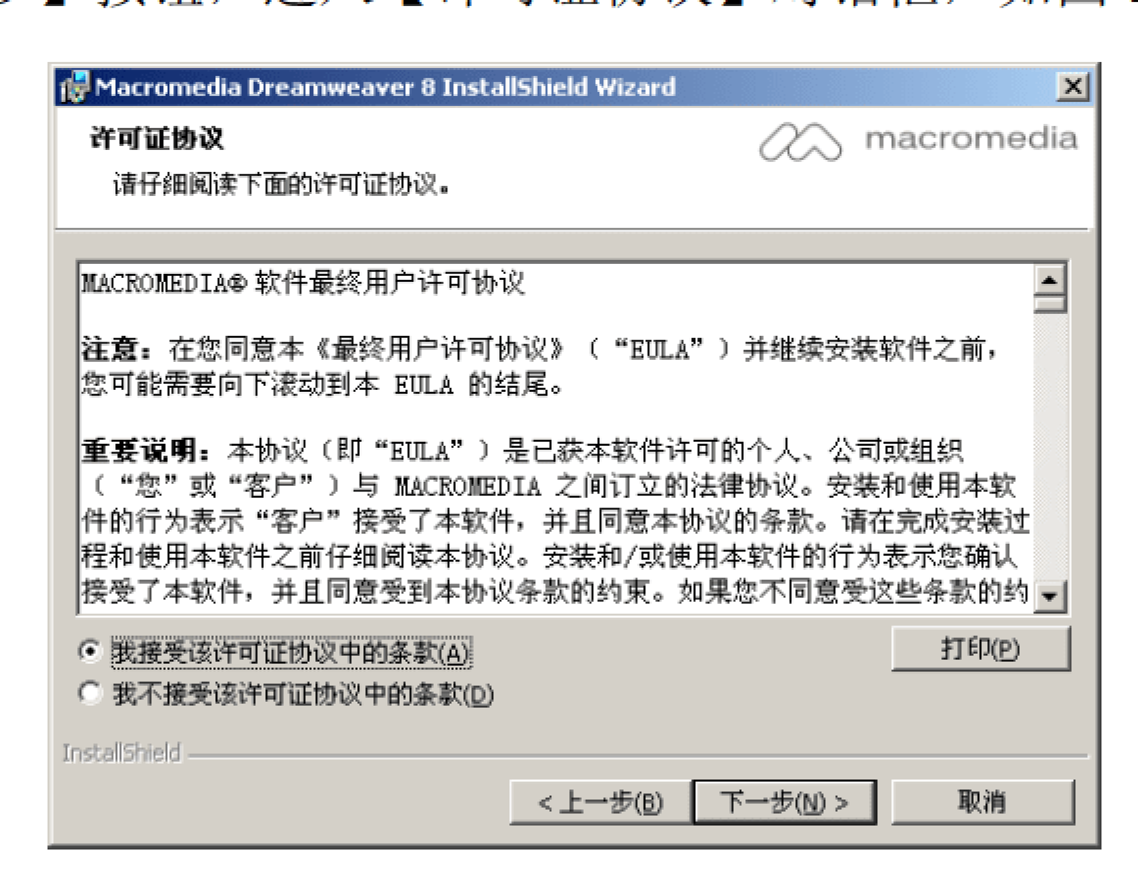


图 1.16　【许可证协议】对话框

(5)　单击【我接受该许可证协议中的条款】单选按钮，单击【下一步】按钮，进入【目标文件夹和快捷方式】对话框，如图 1.17 所示。

(6)　选择文件所要安装到的磁盘目录，并选择是否需要创建快捷方式。单击【下一步】按钮，进入【默认编辑器】对话框，如图 1.18 所示。在这里，可以将 Dreamweaver 8 与某些文件类型进行关联，以便在双击这些类型的文件时直接调用 Dreamweaver 8 进行编辑。

(7)　单击【下一步】按钮，进入【已做好安装程序的准备】对话框，如图 1.19 所示。

图 1.17　【目标文件夹和快捷方式】对话框

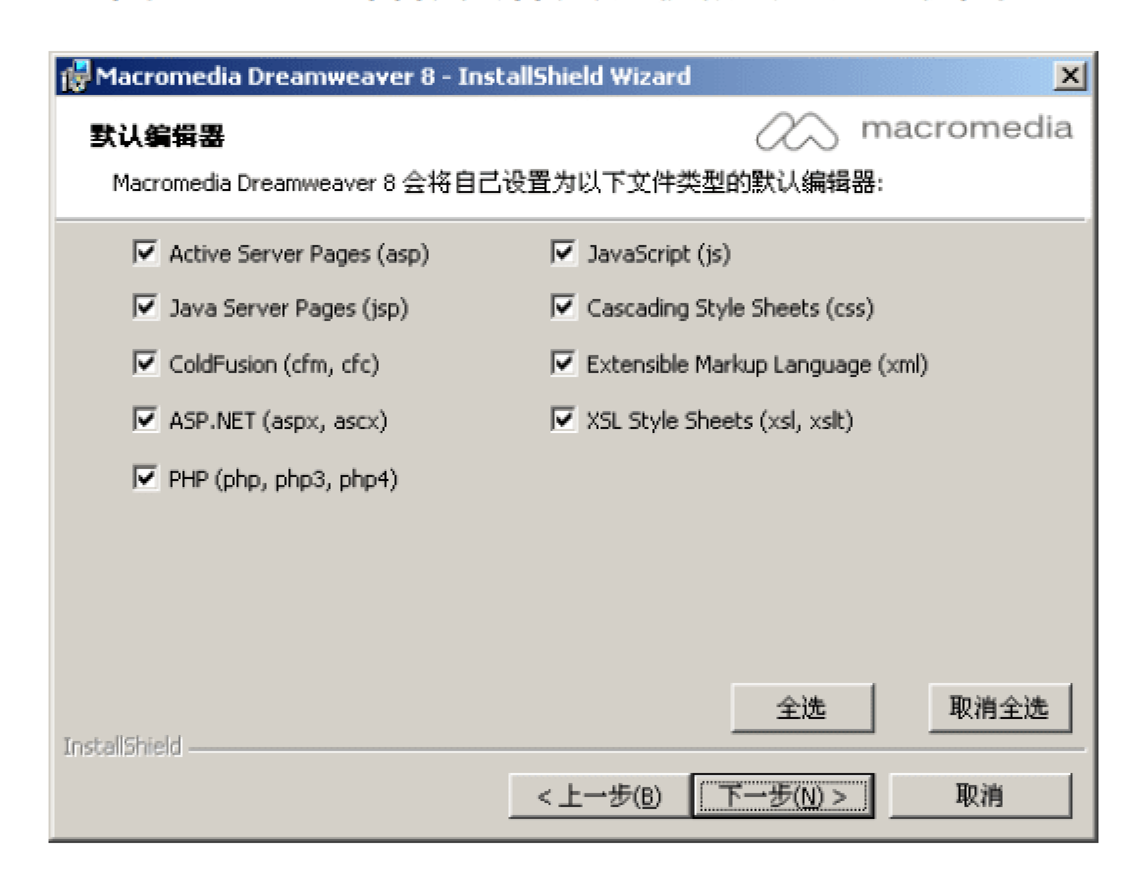


图 1.18　【默认编辑器】对话框

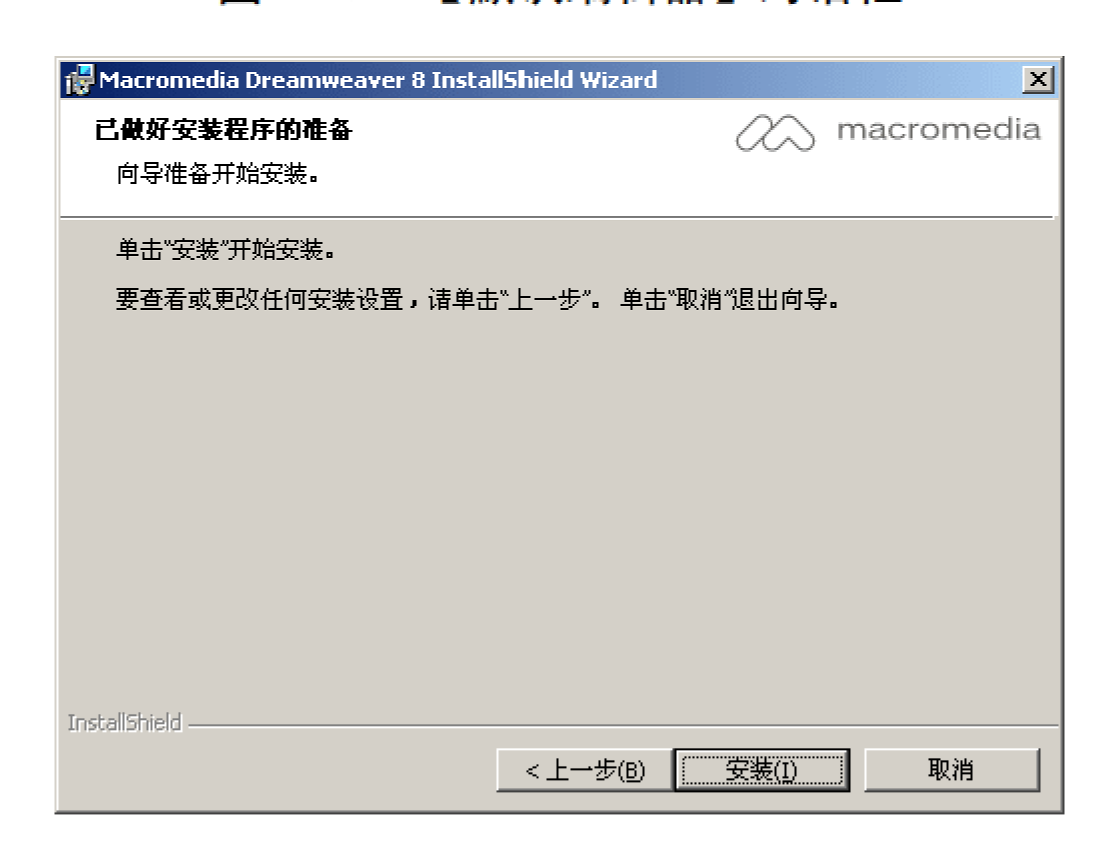


图 1.19　【已做好程序安装的准备】对话框

(8)　此时，安装前的设置工作已全部完成。如不再更改任何安装设置，则单击【安装】按钮，进入【正在安装】对话框，如图 1.20 所示。

(9)　Dreamweaver 8 的安装过程一般包括验证安装、复制新文件、更新组件注册表等几个阶段。此时，用户无须进行操作，安装程序将自动完成。安装完成后，将自动进入【安装向导完成】对话框，如图 1.21 所示。

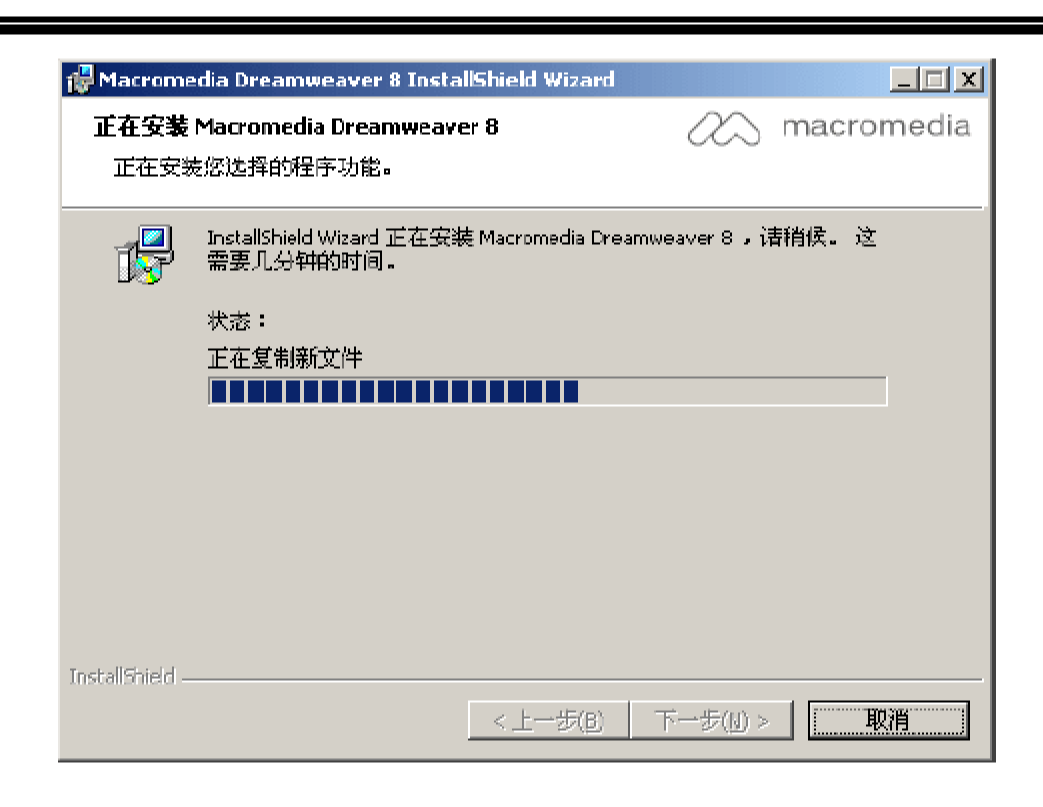


图 1.20　【正在安装 Dreamweaver 8】对话框

图 1.21　InstallShield Wizard 完成对话框

(10) 在此对话框中，用户可确定是否在安装完成后启动 Macromedia Dreamweaver 8 或显示自述文件。Dreamweaver 8 的自述文件如图 1.22 所示。

图 1.22　Dreamweaver 8 的自述文件

(11) 单击【完成】按钮，结束安装。至此，已经全部完成 Dreamweaver 8 的安装。

1.3　Dreamweaver 8 的工作区

安装 Dreamweaver 8 后，就可以通过 Dreamweaver 8 创建自己的 Web 站点了。在使用 Dreamweaver 8 之前，首先认识一下 Dreamweaver 8 的工作区。

Dreamweaver 8 的工作区，是用户操作的主要界面。如图 1.23 所示，Dreamweaver 8 提供了一个将全部元素置于一个窗口中的集成布局。在集成的工作区中，全部窗口和面板都被集成到一个更大的应用程序窗口中。

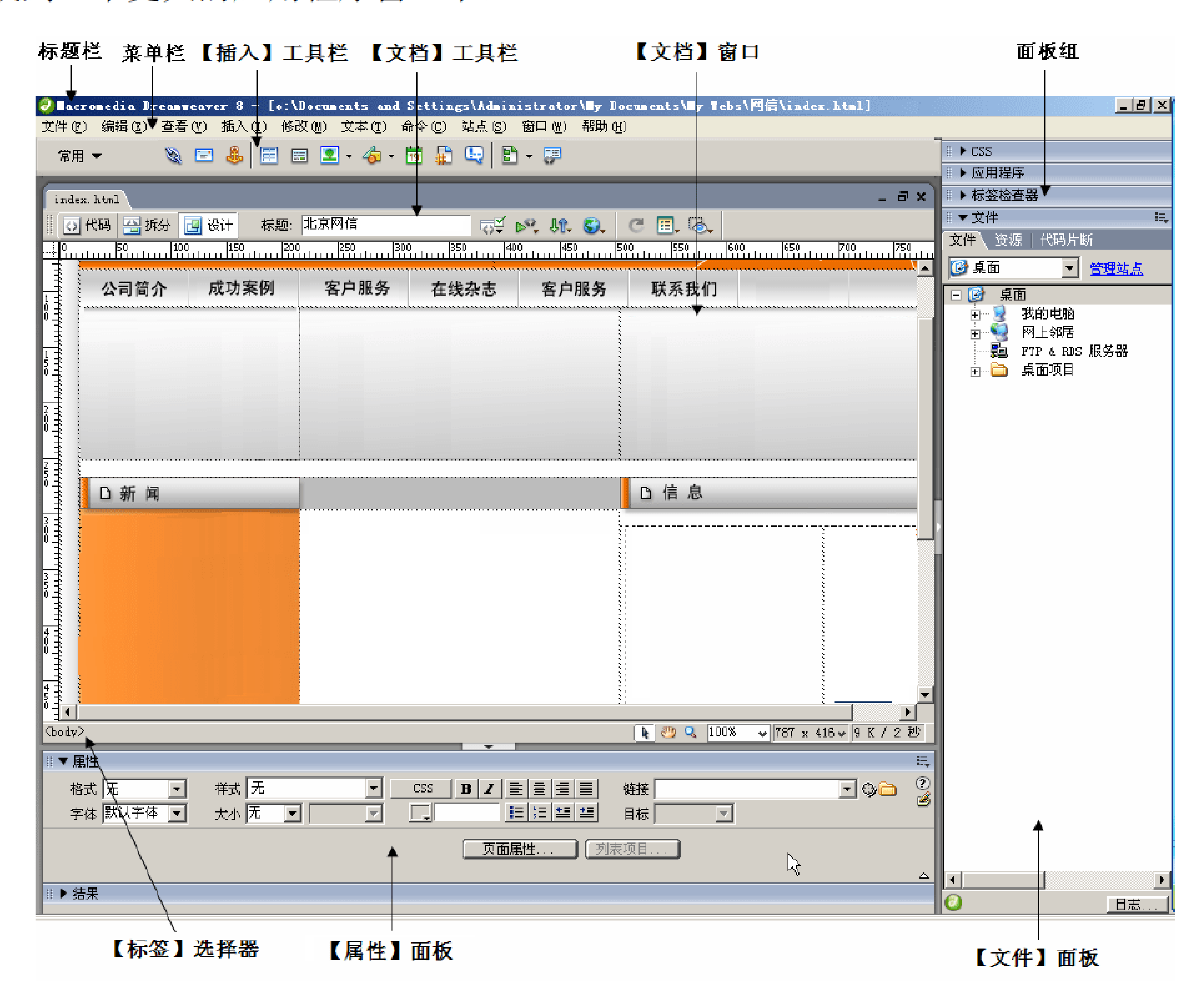


图 1.23　Dreamweaver 8 的工作区

1.3.1　【插入】工具栏

【插入】工具栏包含用于将各种类型的对象(如表格、图形、链接等)插入到文档中的按钮。当将鼠标移到一个按钮上时，将会出现一个工具提示，即该按钮的名称。在 Dreamweaver 8 中，各种对象的按钮分别被组织到不同的类别中，可以通过单击【插入】工具栏左侧的当前类别按钮，如 常用 ▼ 按钮进行切换。

默认情况下，【插入】工具栏包含以下类别。

- 【常用】类别：包含了最常用的对象按钮，如表格、图形等。
- 【布局】类别：包含了用于文档布局的对象按钮，如表格、DIV 标签、层、框架等。
- 【表单】类别：包含了用于创建表单和插入表单元素的对象按钮，如表单、复选

框、单选按钮、下拉列表框等。

- 【文本】类别：包含了各种文本格式设置标签和列表格式设置标签，如加粗、斜体、标题、项目列表、编号列表等。
- HTML 类别：包含了常用的 HTML 标签，如水平线、文件头、表格、框架以及脚本等。
- 【应用程序】类别：包含了用于插入各种动态元素的对象按钮，如记录集、重复区域、用户身份验证、XSL 转换等。
- 【Flash 元素】类别：包含了用于插入 Flash 元素的对象按钮。
- 【收藏夹】类别：用于将【插入】工具栏中最常用的按钮分组和组织到某一公共位置，以方便用户使用。

此外，如果当前文档包含服务器代码时(如 ASP.NET 文档)，【插入】工具栏还会显示其他相应的类别，如 ASP.NET 类别，如图 1.24 所示。

图 1.24　ASP.NET 类别

1.3.2　【标准】工具栏

【标准】工具栏包含与【文件】和【编辑】菜单中的常用命令功能相同的按钮，如【新建】、【打开】、【保存】、【全部保存】、【打印代码】、【剪切】、【复制】、【粘贴】、【撤消】、【重做】等。

1.3.3　【文档】工具栏

【文档】工具栏包含了用于在文档的不同视图之间进行快速切换的相关按钮，以及各种查看选项和一些与常用操作相关的按钮(如验证标记、文件管理、在浏览器中预览/调试等)。

1.3.4　面板组

面板组是组合在一个标题下面的相关面板的集合。通过单击面板组名称左侧的展开/折叠按钮，可展开/折叠所有面板组。如果需要关闭某个面板组，可右击该面板组，从弹出的快捷菜单中选择【关闭面板组】命令。

1.3.5　【标签】选择器

【标签】选择器通常位于文档窗口底部的状态栏中，它显示了环绕当前选定内容的标签的层次结构。单击该层次结构中的任何标签，可选择该标签及其全部内容。

1.3.6　【属性】面板

【属性】面板用于查看和编辑当前文档中所选择的页面元素(如插入的对象或文本)的常用属性。所选的页面元素不同，【属性】面板中的选项也有所不同，有时差别很大。

1.3.7　文档窗口

文档窗口显示了当前创建和编辑的文档。

1.3.8　【文件】面板

【文件】面板用于管理文件和文件夹，包括 Dreamweaver 8 站点中的文件、远程服务器上的文件以及本地磁盘中的所有文件。

1.4　自定义 Dreamweaver 8 开发环境

在使用 Dreamweaver 8 时，可以通过设置一些选项来自定义 Dreamweaver 8 开发环境，以使其更加适合我们的需要。

1.4.1　选择工作区布局

在 1.1 节中，提到 Dreamweaver 8 提供了设计器和编码器两种工作区布局，同时在首次启动 Dreamweaver 8 时将会弹出一个对话框让用户选择一种工作区布局。那么，当用户选择了一种工作区布局后又要切换到另一种工作区布局，又该如何操作呢？

选择【窗口】|【工作区布局】命令，然后选择所要切换的工作区布局，如图 1.25 所示。

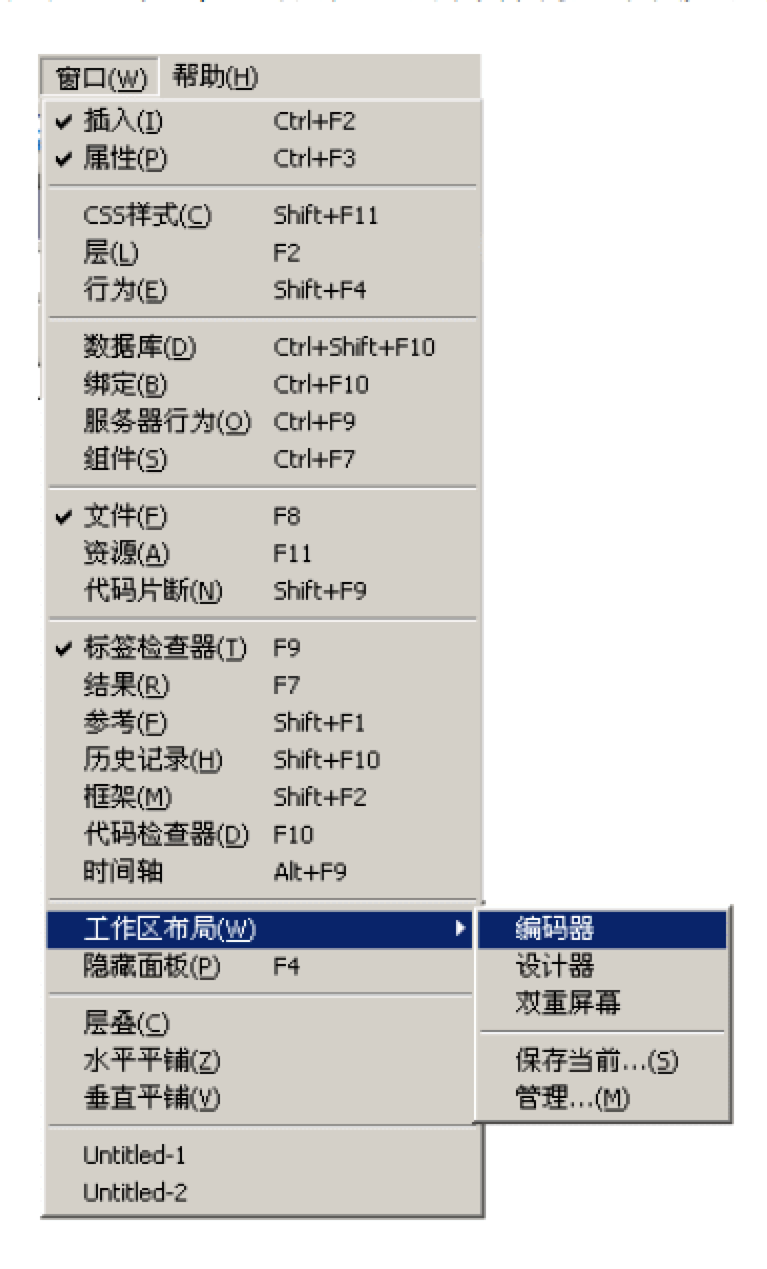


图 1.25　选择工作区布局

在可选择的工作区布局命令中，有一个【双重屏幕】命令。如果此时用户的计算机连接了两个显示器，则可在一个显示器中显示【代码】视图，另一个显示器中显示【设计】视图；如果仅连接了一个显示器，则【代码】视图和【设计】视图将分为两个窗口显示。

1.4.2　隐藏和显示起始页

一般情况下，在启动 Dreamweaver 8 时以及在每次没有打开任何文档时，都会显示 Dreamweaver 8 起始页，如图 1.26 所示。

图 1.26　Dreamweaver 8 起始页

如果在打开 Dreamweaver 8 时不再让其显示起始页，可选中起始页上的【不再显示此对话框】复选框。

如果在隐藏起始页后，又需要显示起始页，则可选择【编辑】|【首选参数】命令，在弹出的【首选参数】对话框的【常规】类别中(如图 1.27 所示)，选中【显示起始页】复选框，单击【确定】按钮即可。

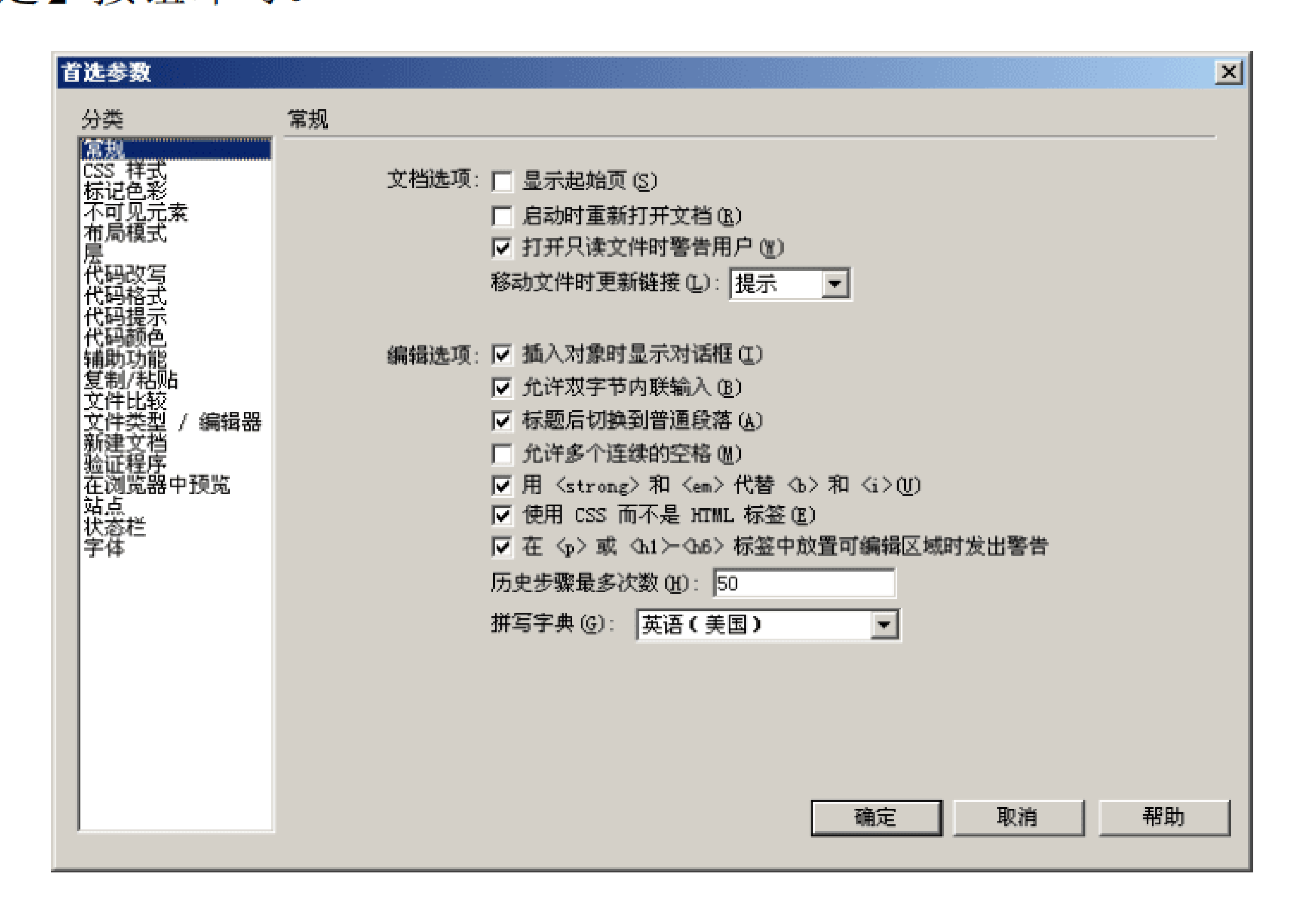


图 1.27　【首选参数】对话框

1.4.3　设置 Dreamweaver 8 的常规外观

如果需要设置 Dreamweaver 8 的常规外观，可选择【编辑】|【首选参数】命令，在弹出的对话框的【分类】列表中选择【常规】类别，如图 1.27 所示。

在此对话框中可设置的首选参数分为两个子类别：【文档选项】和【编辑选项】。这里着重介绍【编辑选项】子类别中的相关选项的作用。

- 【插入对象时显示对话框】复选框用来设置当使用【插入】工具栏或【插入】菜单插入图像、表格、Flash 对象或其他某些对象时，是否弹出对话框以提示输入附加的属性信息。如果取消对此选项的选择，也可通过【属性】面板来设置插入对象的相关属性。
- 【允许双字节内联输入】复选框用来设置是否允许直接将双字节文本输入到【文档】窗口中。如果禁用此功能，则在输入双字节文本时将显示一个用于输入和转换双字节文本的文本输入窗口，文本在被接受后方显示在【文档】窗口中。
- 【标题后切换到普通段落】复选框用来设置在【设计】视图中当在一个标题段落的结尾按下 Enter 键时，是否切换到普通段落。如果选中该选项，则在标题结尾按下 Enter 键后将创建一个用<p>标签标记的新段落。如果取消对该选项的选择，则按 Enter 键后将创建一个用同一标题标签进行标记的新段落。
- 【允许多个连续的空格】复选框用来设置在【设计】视图中输入两个或更多的空格时是否不中断的空格。如果取消对该选项的选择，则所输入的多个空格将被当作单个空格进行处理。
- 【用<strong>和<em>代替<b>和<i>】复选框用来设置，当设置文本为粗体或斜体时，是否使用<strong>标签和<em>标签来代替<b>标签和<i>标签。
- 【使用 CSS 而不是 HTML 标签】复选框用来设置，当使用【属性】面板设置文本格式时是否使用 CSS 样式而不使用 HTML 标签。默认情况下，Dreamweaver 8 会使用 CSS 样式来设置文本格式，即在每次设置所选文本的字体、大小或颜色时，均会创建一个特定于该文档的新样式。如果取消对该选项的选择，Dreamweaver 8 将使用 HTML 标签来设置文本的相关属性。
- 【在<p>或<h1>—<h6>标签中放置可编辑区域时发出警告】复选框用来设置 Dreamweaver 8 在保存一个段落或标题标签内具有可编辑区域的 Dreamweaver 8 模板时是否提示警告信息，该信息会提示：将无法在此区域中创建更多段落。

1.4.4　设置 Dreamweaver 8 的字体显示

根据个人的喜好，可以自行设置在 Dreamweaver 8 的代码窗口中相关文字所使用的字体及大小。如需执行此操作，可选择【编辑】|【首选参数】命令，在弹出的对话框中选择【字体】类别，如图 1.28 所示。

- 【字体设置】列表框用于指定在 Dreamweaver 8 中针对使用给定编码类型的文档所用的字体集。
- 【均衡字体】下拉列表框用于设置普通文本(如段落、标题)所使用的字体。
- 【固定字体】下拉列表框用于设置在<pre>、<code>和<tt>标签内的文本的字体。

● 【代码视图】下拉列表框用于设置在代码视图中的所有文本的字体。

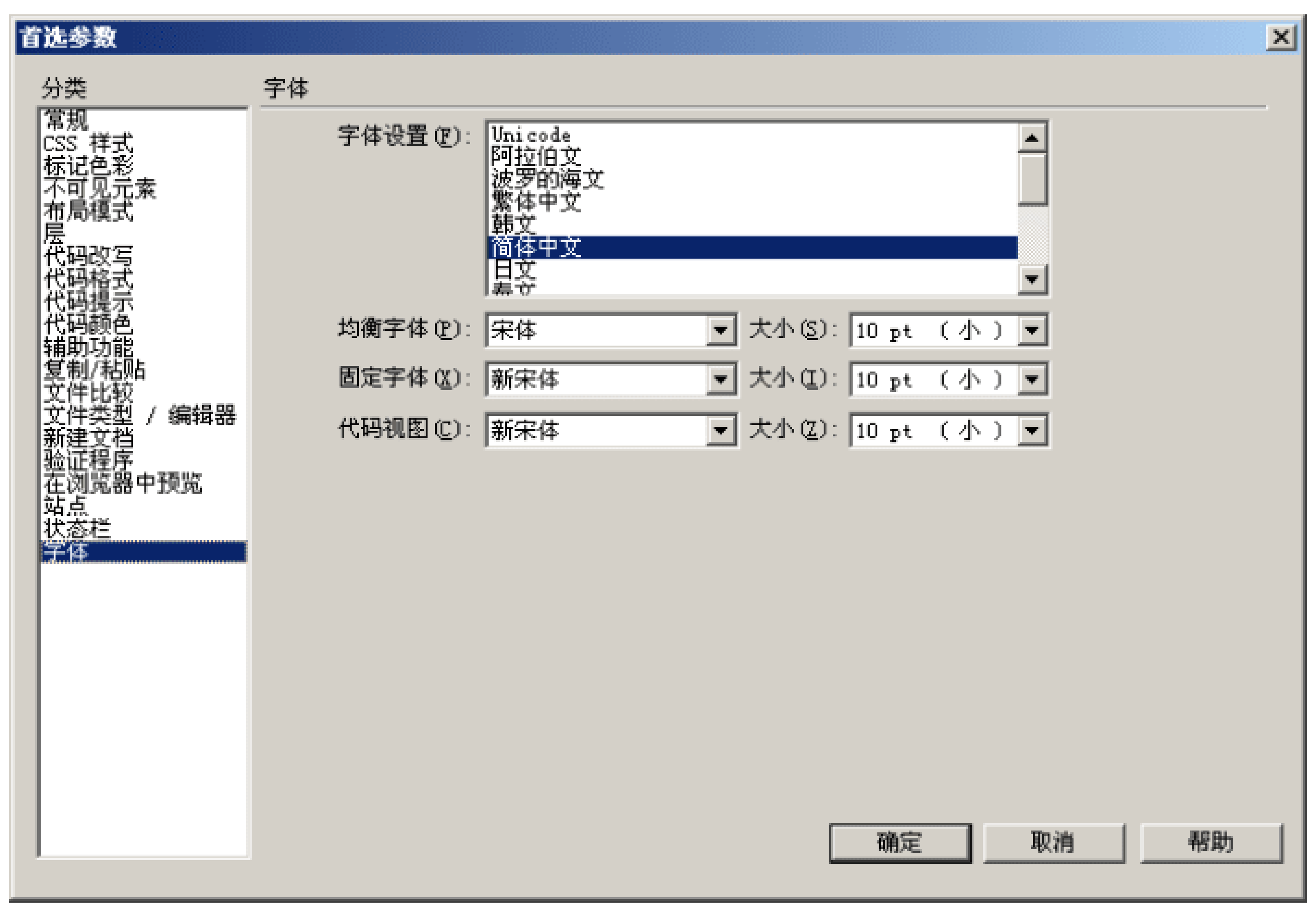


图 1.28 【字体】类别

1.4.5 自定义快捷键

在 Dreamweaver 8 中，根据个人的操作习惯，可以自定义各项操作的快捷键，包括新建快捷键和编辑现有的快捷键。

选择【编辑】|【快捷键】命令，即可打开【快捷键】对话框，如图 1.29 所示。

如要添加快捷键，可首先在【命令】下拉列表框中选择一个命令类别，然后在其下面的列表框中选择所要设置快捷键的命令，单击[+]按钮，然后在【按键】文本框中按下一个组合键，单击【更改】按钮即可。

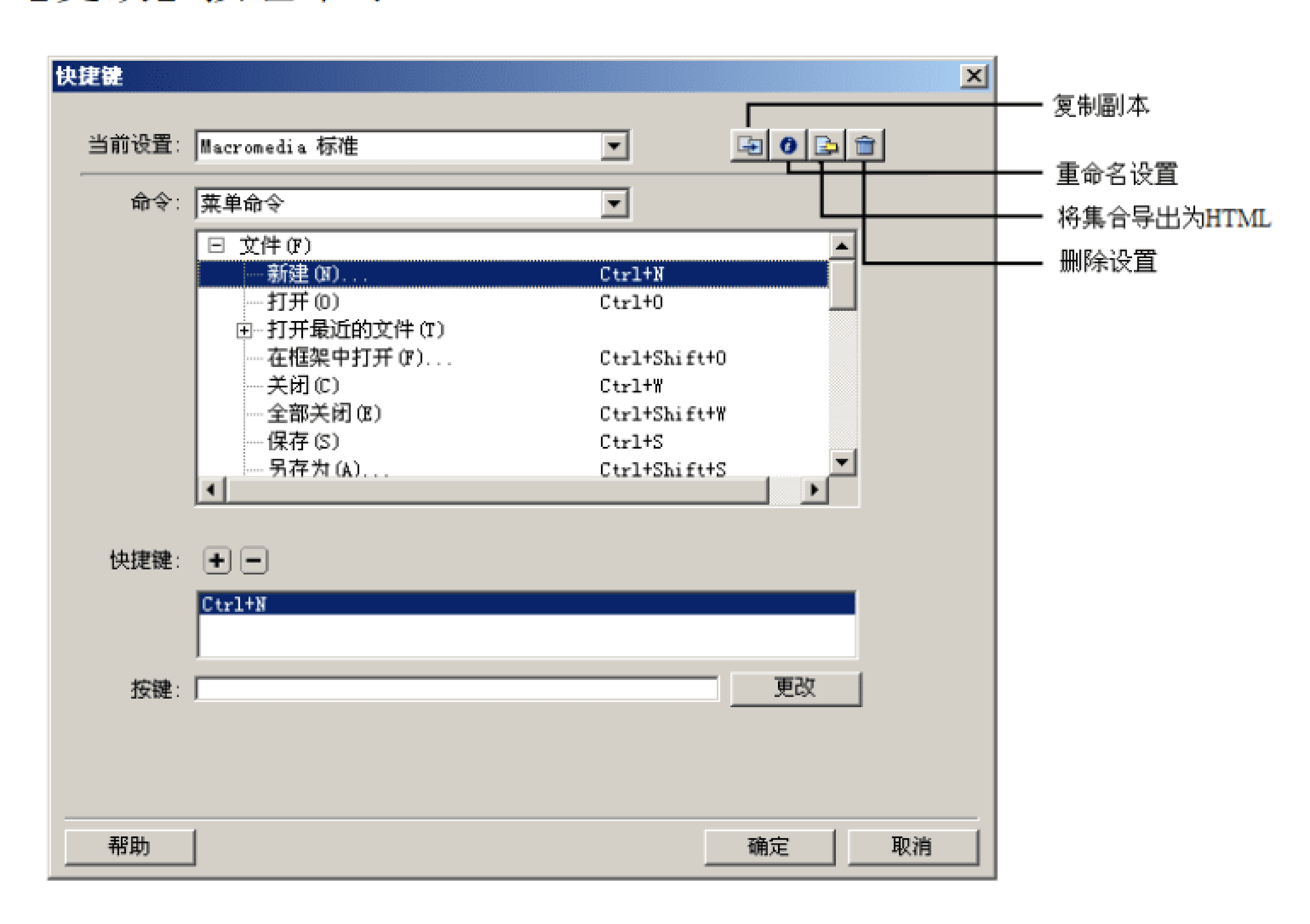


图 1.29 【快捷键】对话框

如要删除一个指定的快捷键，可在【命令】列表框中选择要删除快捷键的命令。此时，该命令的快捷键将出现在【快捷键】列表框中。选择要删除的快捷键，单击[−]按钮即可。

如要编辑现有的快捷键，可在【命令】列表框中选择要编辑快捷键的命令。此时，该

命令的快捷键将出现在【快捷键】列表框中。选择要编辑的快捷键，在【按键】文本框中按下一个新组合键，单击【更改】按钮即可。

需要注意的是，对于 Dreamweaver 8 预定义快捷键组中所设置的快捷键是不能被删除、添加和修改的。此时，可通过单击按钮，来创建一个当前所选择命令的快捷键组的一个副本，然后在【当前设置】下拉列表框中选择该副本，进而在该副本上进行删除、添加和修改。单击按钮，可对当前所选择的快捷键组执行重命名操作；单击按钮，可将当前所选择的快捷键组导出至 HTML 文件；单击按钮，可删除当前所选择的快捷键组。

1.5　典型实例：通过 Dreamweaver 8 创建 ASP.NET 页面

本节将通过 Dreamweaver 8 来创建一个简单的 ASP.NET 页面，使读者对 Dreamweaver 8 和 ASP.NET 的结合使用有一个理性的认识。

(1)　启动 Dreamweaver 8，选择【文件】|【新建】命令，此时将弹出【新建文档】对话框，如图 1.30 所示。

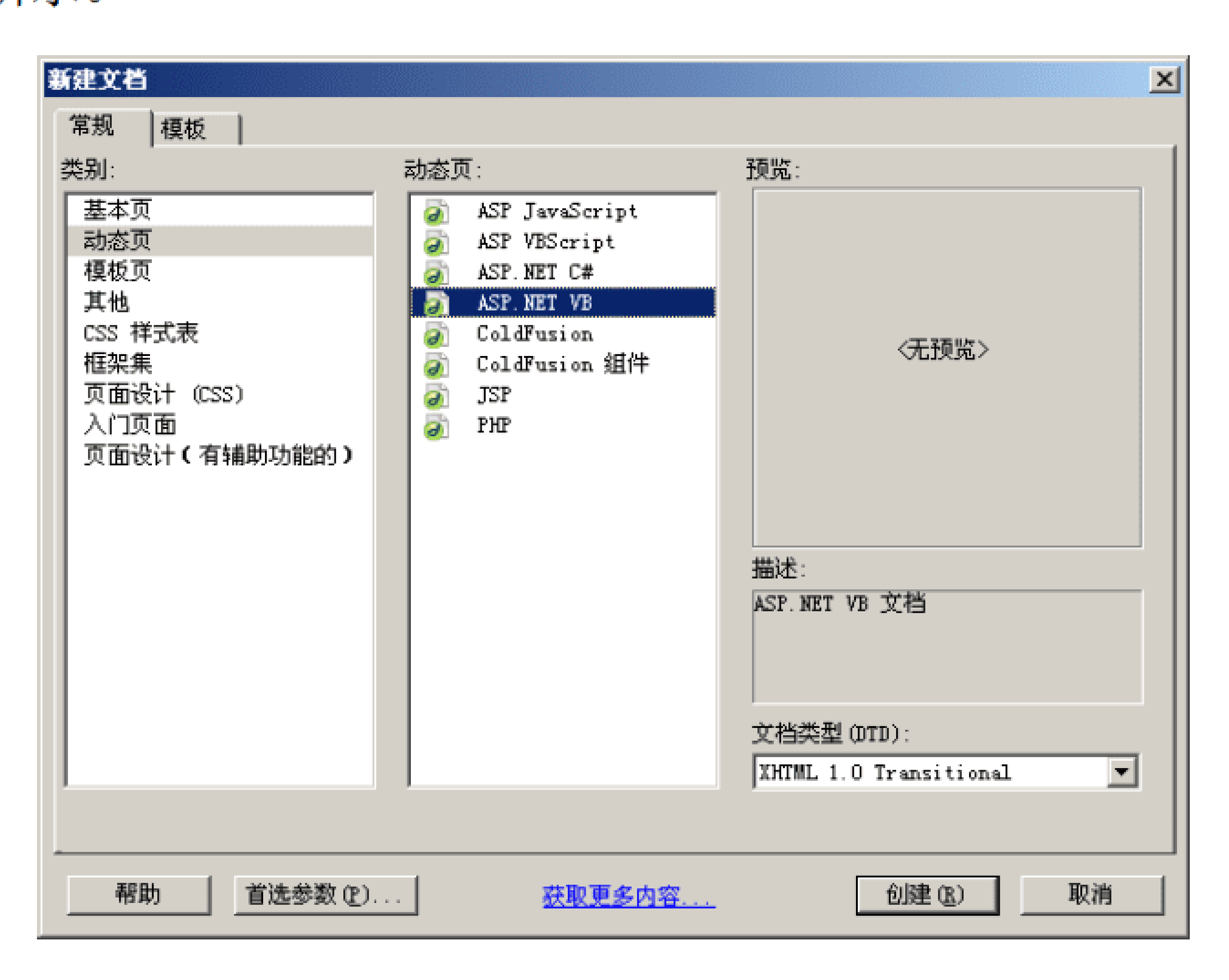


图 1.30　【新建文档】对话框

(2)　在【类别】列表框中，选中【动态页】，并在右侧的【动态页】列表框中选择 ASP.NET VB。单击【创建】按钮，即可新建一个以 VB.NET 为脚本的 ASP.NET 页面。

(3)　将视图切换至【设计】视图，单击【属性】面板中的【页面属性】按钮，在弹出的【页面属性】对话框中，选择【外观】类别，并设置页面字体的大小为 12px，背景颜色值为#CCCCCC，上边距为 10px，如图 1.31 所示。

(4)　在【页面属性】文本框中，提供了对页面的布局和基本格式的设置，包括页面的默认字体、字体大小、文本颜色、背景颜色、背景图像以及边距设置等。单击【确定】按钮，完成页面属性的设置。

(5)　在网页编辑页面中，输入文本“请随意输入内容”，然后选择该段文本，单击【属性】面板中的按钮，将文本居中显示。然后将【插入】工具栏中的类别切换至 ASP.NET，拖动按钮至刚才输入的文本之后。在弹出的【asp:文本框】对话框中，设置 ID 为 TextBox1，

【工具提示】为“请随意输入内容”，如图 1.32 所示。单击【确定】按钮，完成文本框控件的插入。

图 1.31　【页面属性】对话框

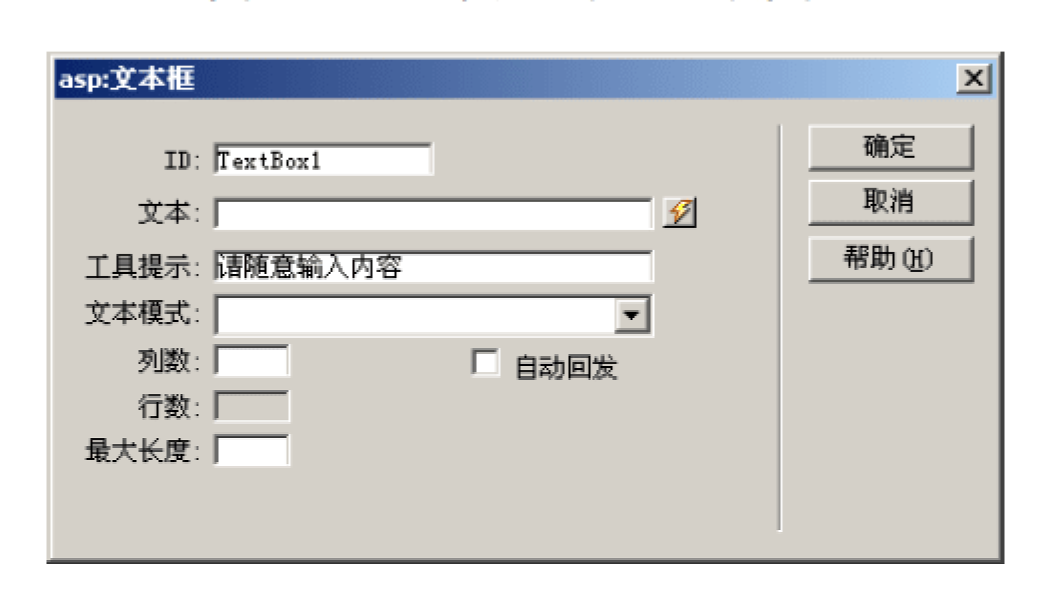


图 1.32　【asp:文本框】对话框

(6) 此时，页面将会伴随着文本框控件，生成一个表单(Form)，同时带有属性"Runat=Server"。选择所创建的文本，并将其拖动至表单(Form)内的文本框控件之前。然后，在文本框控件之后，按 Enter 键换一行，单击【ASP.NET-插入】工具栏中的按钮，在弹出的【asp:按钮】对话框中，设置其 ID 为 Button1，【文本】为“点击试试”，如图 1.33 所示。

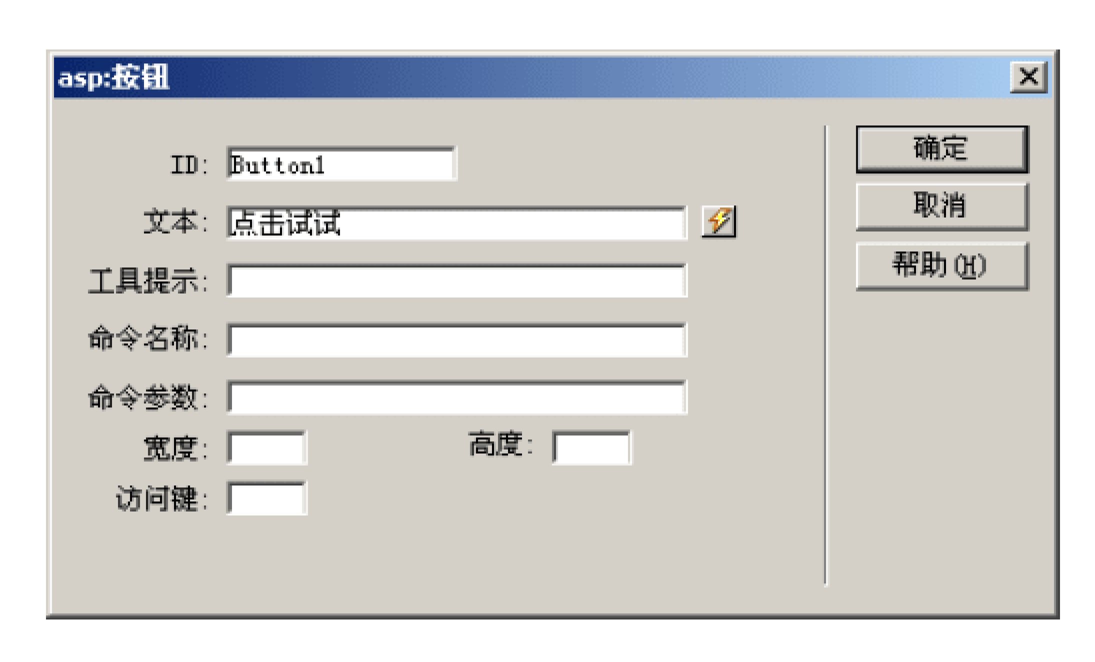


图 1.33　【asp:按钮】对话框

(7) 单击【确定】按钮，此时在页面上将添加一个按钮。选择该按钮控件并右击，从弹出的快捷菜单中选择【编辑标签】命令，在弹出的【标签编辑器】对话框中选择【事件】类别下的 OnClick，并在右侧的文本框中输入 Button1_Click，如图 1.34 所示。

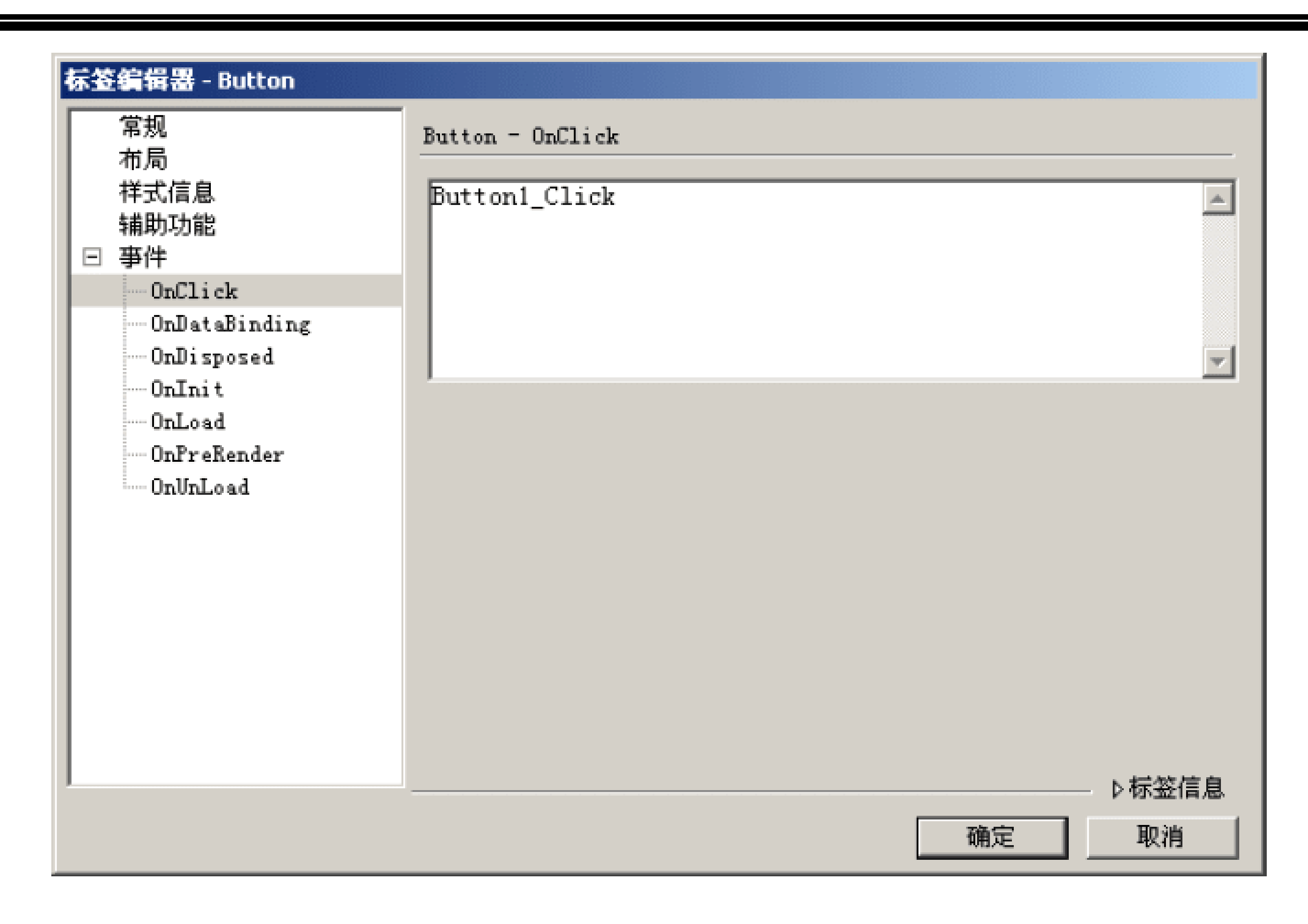


图 1.34　【标签编辑器】对话框

(8) 单击【确定】按钮，完成按钮单击事件的设置。步骤(7)指定了当单击该按钮时所执行的事件名称。对于该事件的具体定义，需要在【代码】视图中进行编写，这将在稍后进行介绍。

(9) 将光标置于按钮控件之后，按 Enter 键换行，然后单击【ASP.NET-插入】工具栏中的 abc 按钮，在弹出的【asp:标签】对话框中，设置 ID 为 LblInfo，如图 1.35 所示。

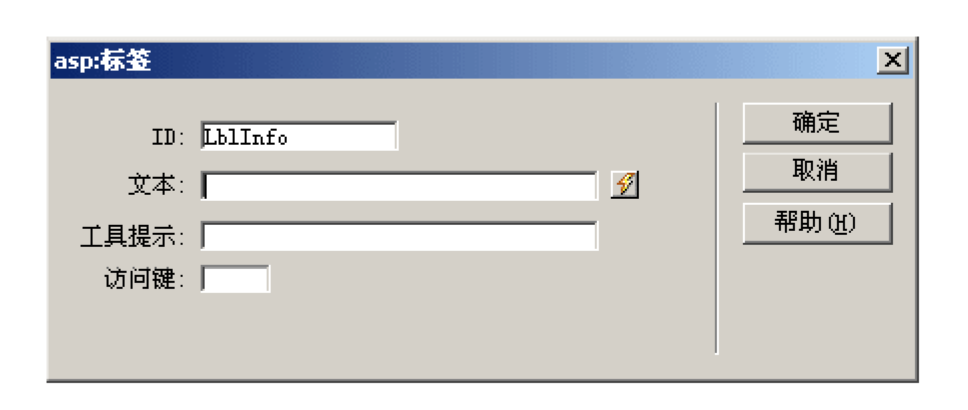


图 1.35　【asp:标签】对话框

(10) 单击【确定】按钮，完成标签控件的插入。选择该标签控件并右击，从弹出的快捷菜单中选择【编辑标签】命令，在弹出的【标签编辑器】对话框中，选择【布局】类别，设置其前景色为红色，如图 1.36 所示。

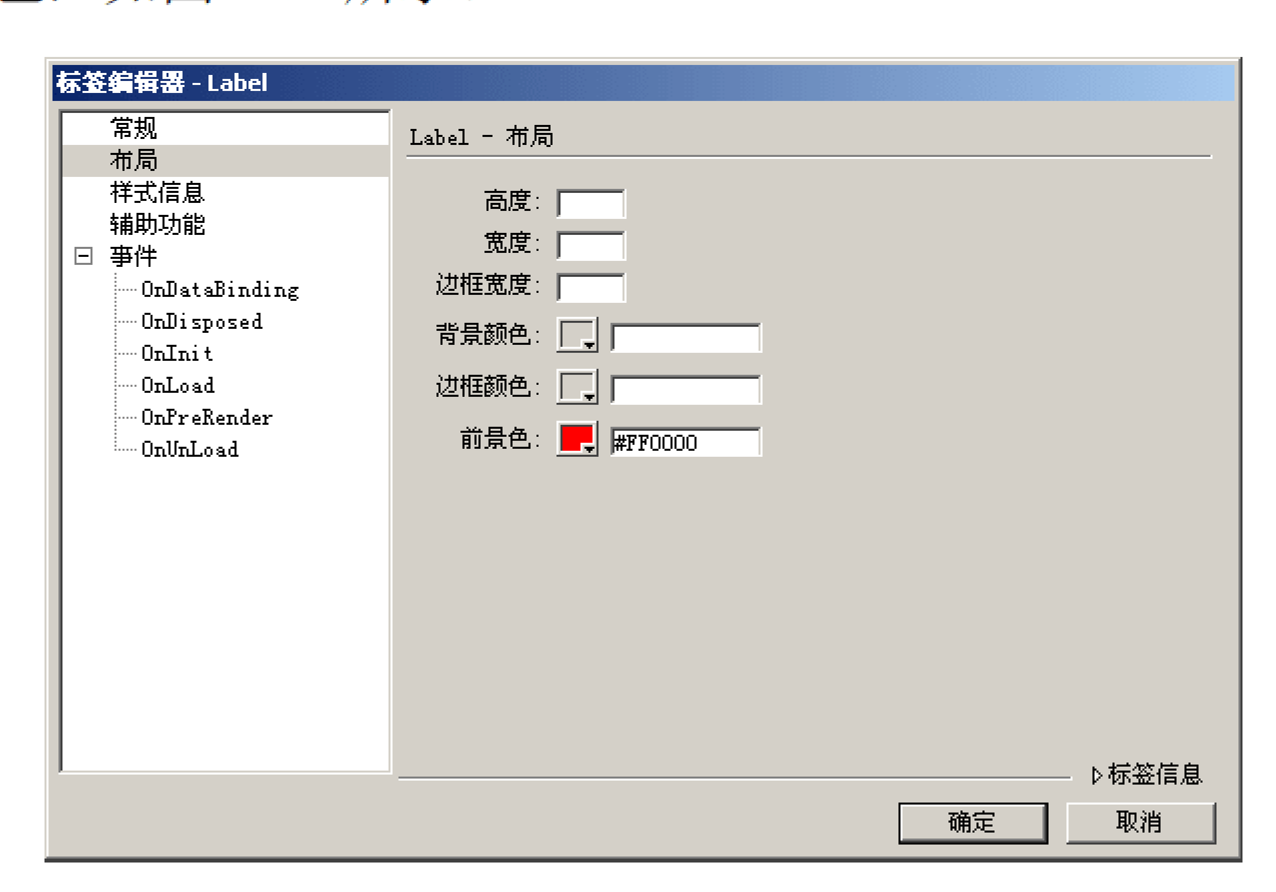


图 1.36　【标签编辑器】对话框

(11) 单击【确定】按钮，完成标签控件的设置。步骤(10)是将标签中的文本的字体颜色设置为红色。

(12) 切换至【代码】视图，为按钮控件添加单击事件 Button1_Click。首先，将光标置于<title>标签之前，单击【ASP.NET-插入】工具栏中的【页面载入】按钮，此时将会在光标所在处生成一段脚本代码，如图 1.37 所示。

```
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<script runat="server">
Sub Page_Load(Src As Object, E As EventArgs)
If Not IsPostBack Then
DataBind()
End If
End Sub
</script>
<title>我的第一个ASP.NET页面</title>
```

图 1.37　自动生成的脚本

(13) 这里所添加的 Page_Load 事件是页面加载时将首先执行的事件代码，其中的 DataBind()过程是 Dreamweaver 在 Page_Load 事件中默认加载的用于绑定数据的自定义过程。在本页面中，暂时不需要 Page_Load 事件，可将其修改为 Button1_Click 事件，并对其代码进行修改，如图 1.38 所示。

```
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<script runat="server">
Sub Button1_Click(Src As Object, E As EventArgs)
  LblInfo.text="您所输入的是: " & TextBox1.Text
End Sub
</script>
<title>我的第一个ASP.NET页面</title>
```

图 1.38　Button1_Click 事件

(14) 然后，选择【文件】|【保存】命令(或按下快捷键 Ctrl+S)，在弹出的【另存为】对话框中，选择 C 盘下面的 Inetpub 目录中的 wwwroot 子目录，并将文件名设置为 MyFirstPage.aspx。

提示： 这里的 C 盘为系统盘(如果您的操作系统是装在其他盘中，请选择相应的系统盘)，而 wwwroot 文件夹是系统在安装 IIS 后自动设置的默认发布目录。

(15) 一个简单的 ASP.NET 页面已经完成。接下来可以进行测试，打开浏览器，在地址栏中输入“http://localhost/myfirstpage.aspx”，按 Enter 键即可打开刚才制作的页面，如图 1.39 所示。

(16) 在文本框中输入文本信息，单击【点击试试】按钮，即可在按钮下方显示用户输入的文本，如图 1.40 所示。

图 1.39　页面浏览

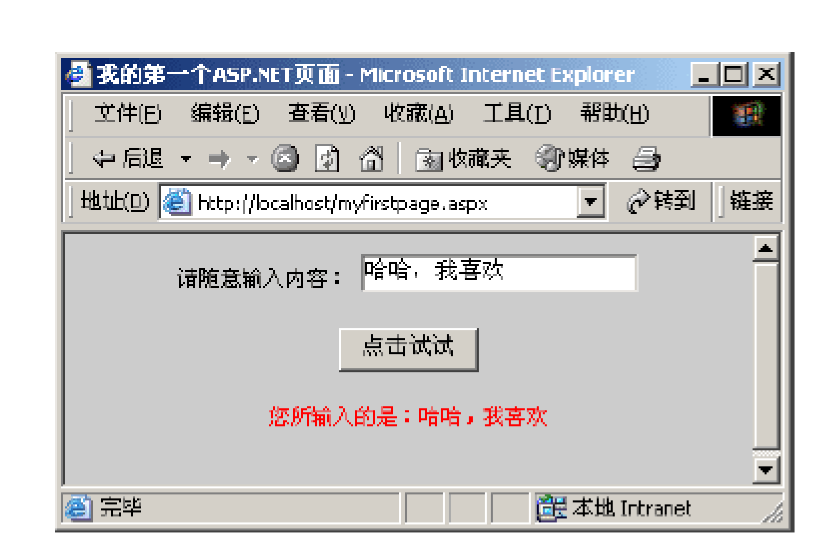


图 1.40　页面浏览

(17) 这里有一个小问题。当在文本框中输入汉字时，在单击【点击试试】按钮后，可能无法正常显示用户所输入的内容，显示的而是一个问号“?”。这是由于 ASP.NET 不支持 gb2312 传输所导致的。解决办法很简单，切换至【代码】视图，将第一行代码中的“ResponseEncoding="gb2312"”去掉即可，如图 1.41 所示。

```
<%@ Page Language="VB" ContentType="text/html" ResponseEncoding="gb2312" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
```

删除

图 1.41　修改代码

1.6　习　　题

(1)　了解 Dreamweaver 8 中各种工具栏的使用。

(2)　了解【ASP.NET-插入】工具栏中各个标签的具体含义。尝试在页面中插入各个标签，了解各个标签相关属性的设置。

第 2 章　站 点 操 纵

众所周知，Dreamweaver 可以对单个的页面文档进行处理。但更多的情况下，Dreamweaver 针对的是整个站点及其中的所有页面。事实上，Dreamweaver 8 不仅提供了单个页面的编辑功能，而且还提供了强大的站点管理功能。

本章将介绍如何通过 Dreamweaver 8 来实现对站点的操纵。

2.1　站 点 概 述

在 Dreamweaver 中，“站点”一词既可以表示 Web 站点，可以表示 Dreamweaver 站点。所谓 Web 站点，是一系列的文档和相关资源的组合。这些文档可以相互链接，也可以毫无关系，而资源则包括文档中所调用的各种图片、Flash 动画、CSS 文件以及数据库等。简单地说，一个网站就是一个 Web 站点。而 Dreamweaver 站点则主要用于管理相应的 Web 站点，它提供了一种组织所有与 Web 站点相关联的文档的方法。通过 Dreamweaver，可以将站点上传到 Web 服务器上，并对其进行自动跟踪、维护链接和管理文件等。

在设置 Dreamweaver 站点之前，首先必须了解以下几个基本概念。

(1)　本地文件夹：又叫本地站点，是用来存储站点文件的工作副本的文件夹，它也是 Dreamweaver 站点的工作目录。本地文件夹并不表示其存储的位置是在本地计算机上，它也可以位于网络服务器上。在 Dreamweaver 中，必须为每一个新创建的 Web 站点定义一个本地文件夹。

(2)　远端文件夹：又叫远端站点，是用户在运行 Web 服务器的远程计算机上用来存储站点文件的工作副本的文件夹。远端文件夹与本地文件夹是相对应的，在最常见的 Dreamweaver 工作流程中，用户在本地文件夹(本地站点)对文件进行创建和编辑操作，然后将其上传到远端文件夹(远程站点)中。

(3)　测试文件夹：所谓测试文件夹，是指 Dreamweaver 用来处理动态页面(如 ASP.NET 页面)的文件夹。Dreamweaver 使用该文件夹生成动态内容并在工作时连接到数据库。一般情况下，测试文件夹与远端文件夹为同一文件夹。

2.2　新 建 站 点

在了解了 Dreamweaver 中的站点定义后，下面来看如何新建一个站点。

在 Dreamweaver 8 中，使用站点定义向导可以快速新建一个站点，其具体步骤如下。

1. 设置站点名称及 HTTP 地址

选择【站点】|【新建站点】命令，此时系统将会弹出站点定义的向导对话框，如图 2.1

所示。

图 2.1　设置站点名称

在该对话框中，首先必须输入站点的名字，这里的名字仅仅是一个代号，用于在 Dreamweaver 8 中标识你所定义的站点。该名字用户可自行定义，无须与站点的文件夹同名。在站点名称下面，可指定站点的 HTTP 地址(URL)。通过该地址，Dreamweaver 可确保站点根目录相对链接可在远端服务器上工作。

2. 确认是否使用服务器技术

设置完站点名称及 HTTP 地址后，单击【下一步】按钮，向导对话框将让你设置是否使用服务器技术，如图 2.2 所示。

图 2.2　设置是否打算使用服务器技术

此对话框主要是询问用户目前所创建的站点是否使用服务器技术，如 ColdFusion、ASP.NET、ASP、JSP、PHP 等。如果要创建的站点为静态网站，则可单击【否，我不想使用服务器技术】单选按钮；如果要创建的站点为动态网站，则可单击【是，我想使用服务器技术】单选按钮，并在下面的下拉列表框中选择相应的服务器语言。

提示： 静态网站与动态网站的区别，并不是指网站具不具备 Flash 动画或 GIF 动画，而是指网站具不具有互动性。所谓互动性，是指网站是否能根据用户的要求或选择作出响应，是否能根据不同的条件链接不同的页面，是否能与数据库进行交互等。

3. 设置开发环境

单击【下一步】按钮，进入下一步向导对话框，如图 2.3 所示。

图 2.3　设置如何使用您的文件

此对话框主要用来设置站点的开发环境，即开发过程中如何对文件进行处理，这里提供了 3 个选项：【在本地进行编辑和测试】、【在本地进行编辑，然后上传到远程测试服务器】和【使用本地网络直接在远程测试服务器上进行编辑】。

如果选择前两项，系统将要求输入文件的存储位置，即站点的工作目录；如果选择第 3 项，系统则会要求输入远程服务器的地址。

提示： 对于初学者，考虑到安全性，建议使用前两种方式来处理站点。

4. 测试文件

由于用户在上一步中会有不同的选择，此步操作也会有不同。这里，以用户选择了【在本地进行编辑和测试】选项为例进行介绍。当用户在第 3 步操作中选择了第 1 项后，单击【下一步】按钮后，系统将弹出如图 2.4 所示的对话框。

这里需要设置的是用于测试的 URL 地址前缀。一般来说，URL 前缀不包括任何文件名，它由域名和 Web 站点的主目录的任何一个子目录或虚拟目录组成。如果 Dreamweaver

与 Web 服务器运行在同一台计算机上，则可使用 localhost 来代替域名。单击【测试 URL】按钮，可测试所设置的 URL 前缀是否正确。

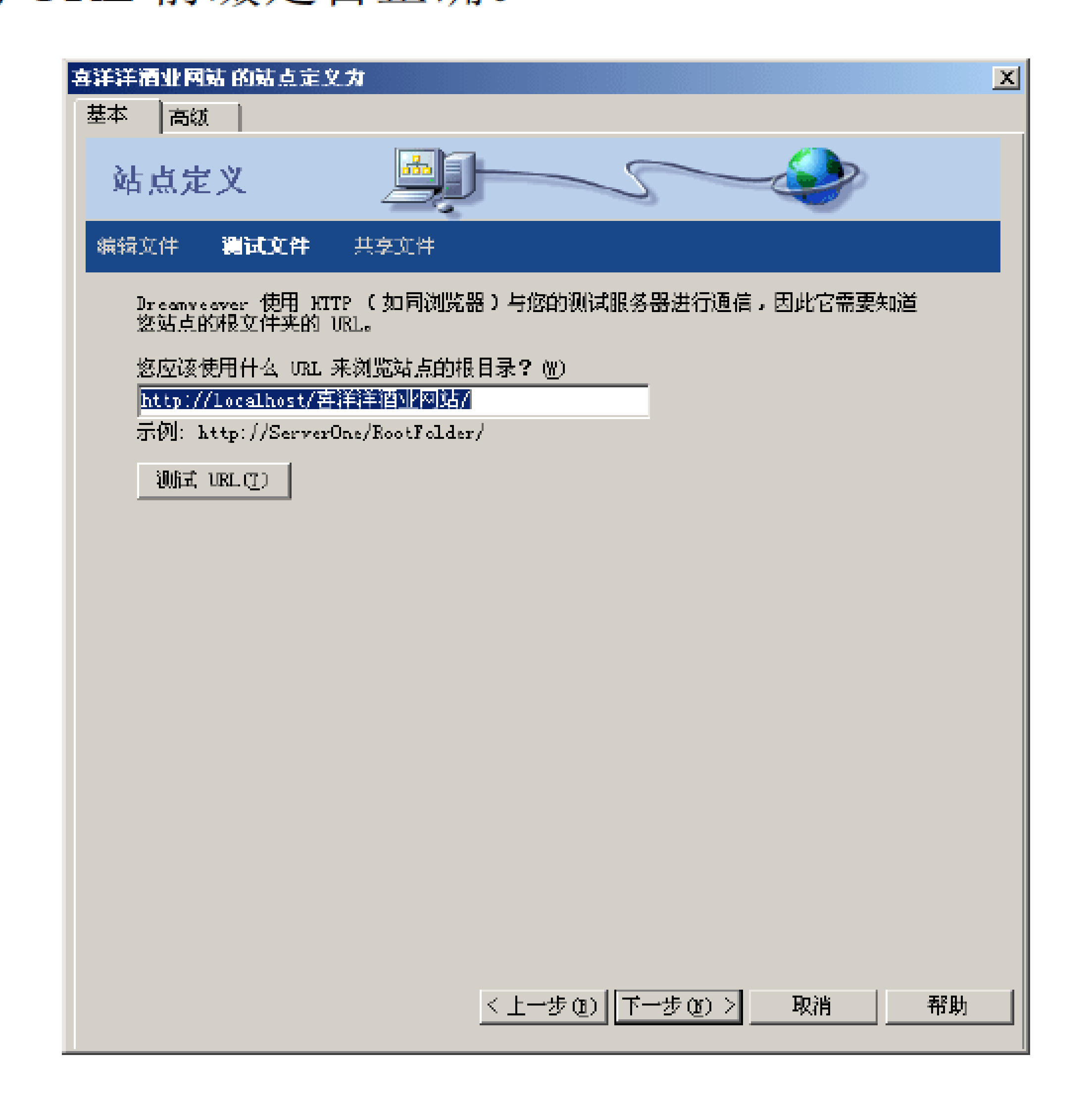


图 2.4　设置测试用的 URL 地址前缀

5. 设置是否使用远程服务器

单击【下一步】按钮，系统将弹出如图 2.5 所示的对话框。

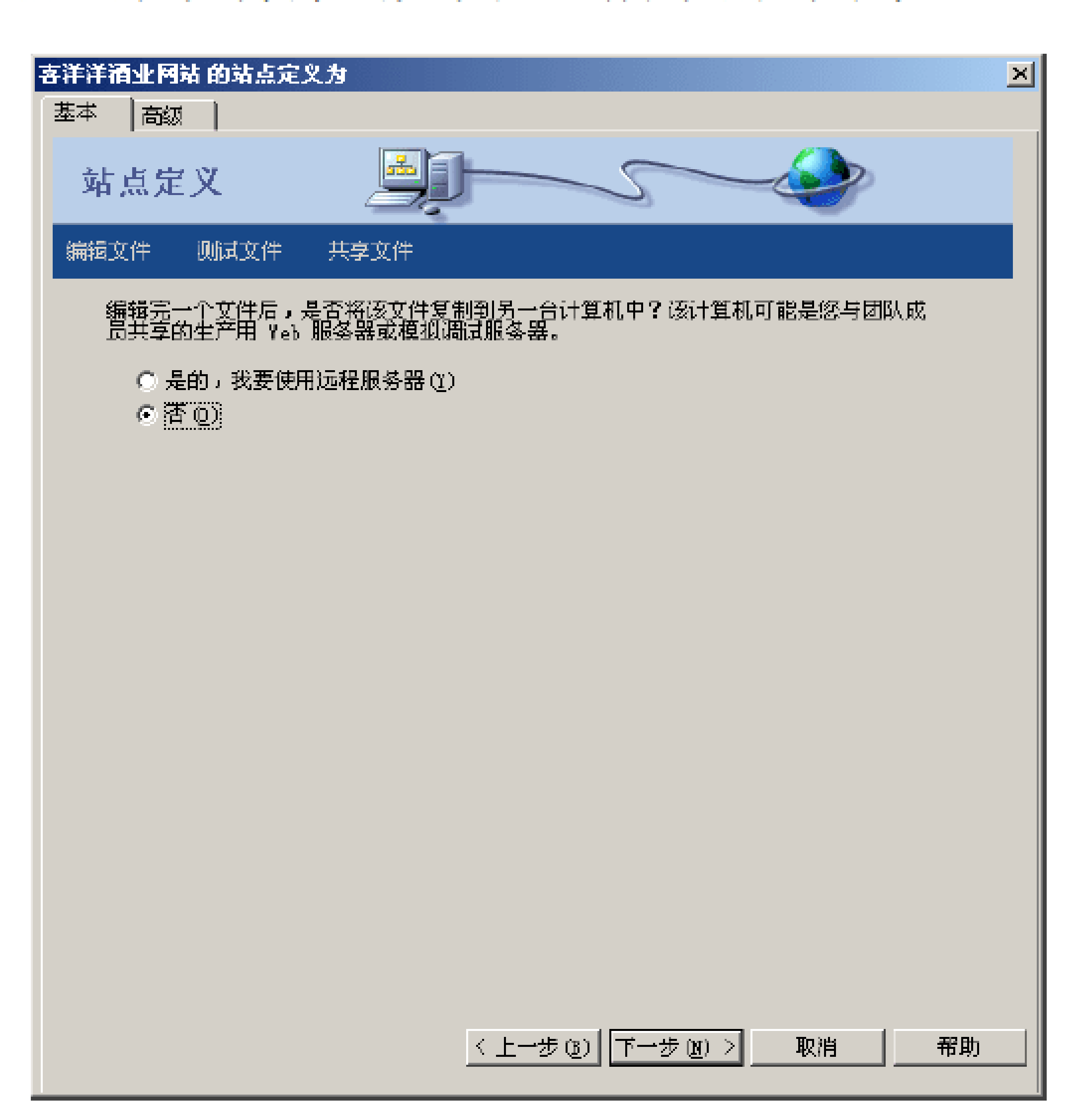


图 2.5　设置是否使用远程服务器

该对话框主要用来设置当编辑完每一个文件后，是否将其复制到远程服务器上。如果选择【否】单选按钮，则系统将跳过第 6 步和第 7 步操作，直接进入第 8 步。

6. 设置如何连接远程服务器

若用户在第 5 步操作中选择了【是的，我要使用远程服务器】单选按钮，则接下来向导将让你对远程服务器进行设置，如图 2.6 所示。

图 2.6　设置如何连接远程服务器

在【您如何连接到远程服务器】下拉列表框中，包括 FTP、【本地/网络】、WebDAV、【SourceSafe 数据库】、RDS 等选项。

- 如果远程服务器为本地计算机或在本地网络中，可选择【本地/网络】选项。
- 如果与远程服务器之间的文件传递是通过 FTP 进行的，则可选择 FTP 选项。此时，还需设置 FTP 地址、存储在服务器上的文件夹名、FTP 用户名、FTP 密码等信息。
- 如果使用 WebDAV(基于 Web 的分布式创作和版本控制)协议与远程服务器连接，则可选择 WebDAV 选项。此时，还需设置 WebDAV 服务器的 URL 地址、WebDAV 用户名与 WebDAV 密码。
- 如果选择【SourceSafe 数据库】选项，则表示通过 Microsoft Visual SourceSafe 与远程服务器进行连接。此时，必须保证服务器上安装了 Microsoft Visual SourceSafe Client。
- 如果选择 RDS 选项，则表示将通过 RDS(远端开发服务)连接到远程服务器上。此时，远端文件夹必须位于运行 ColdFusion 的计算机上。

7. 存回和取出文件设置

单击【下一步】按钮，向导将询问是否启用存回和取出文件，如图 2.7 所示。

如果是在协作环境中工作(或者在多台计算机上独自工作)，则可启用存回和取出文件功能。此功能可防止他人编辑自己目前正在编辑的文件，以确保文件的完整性。其中，“取出”是指调用远程服务器上的文件副本并覆盖本地文件；“存回”是指将本地文件存回，使用本地文件副本覆盖远程文件。当取出文件时，该文件在服务器上将处于不可用状态，其他人也无法使用 Dreamweaver 编辑该文件；当存回文件时，该文件在服务器上将恢复可用状态，其他人可继续使用、取出和编辑。一般情况下，无须启用存回和取出功能。

图 2.7　存回和取出文件设置

8. 站点设置总结

在此步操作中，系统将总结并显示前面用户所设置的各项信息，如图 2.8 所示。

图 2.8　站点设置总结

如果设置没有问题，则可单击【完成】按钮完成新建站点操作；否则，单击【上一步】按钮，返回相应的设置进行修改。

至此，新建站点操作全部完成。如果新建站点的本地文件夹内已有文件，则系统将会立刻启用文件缓存，并将站点内的所有文件导入至【文件】面板中，以便用户操作，如图 2.9 所示。

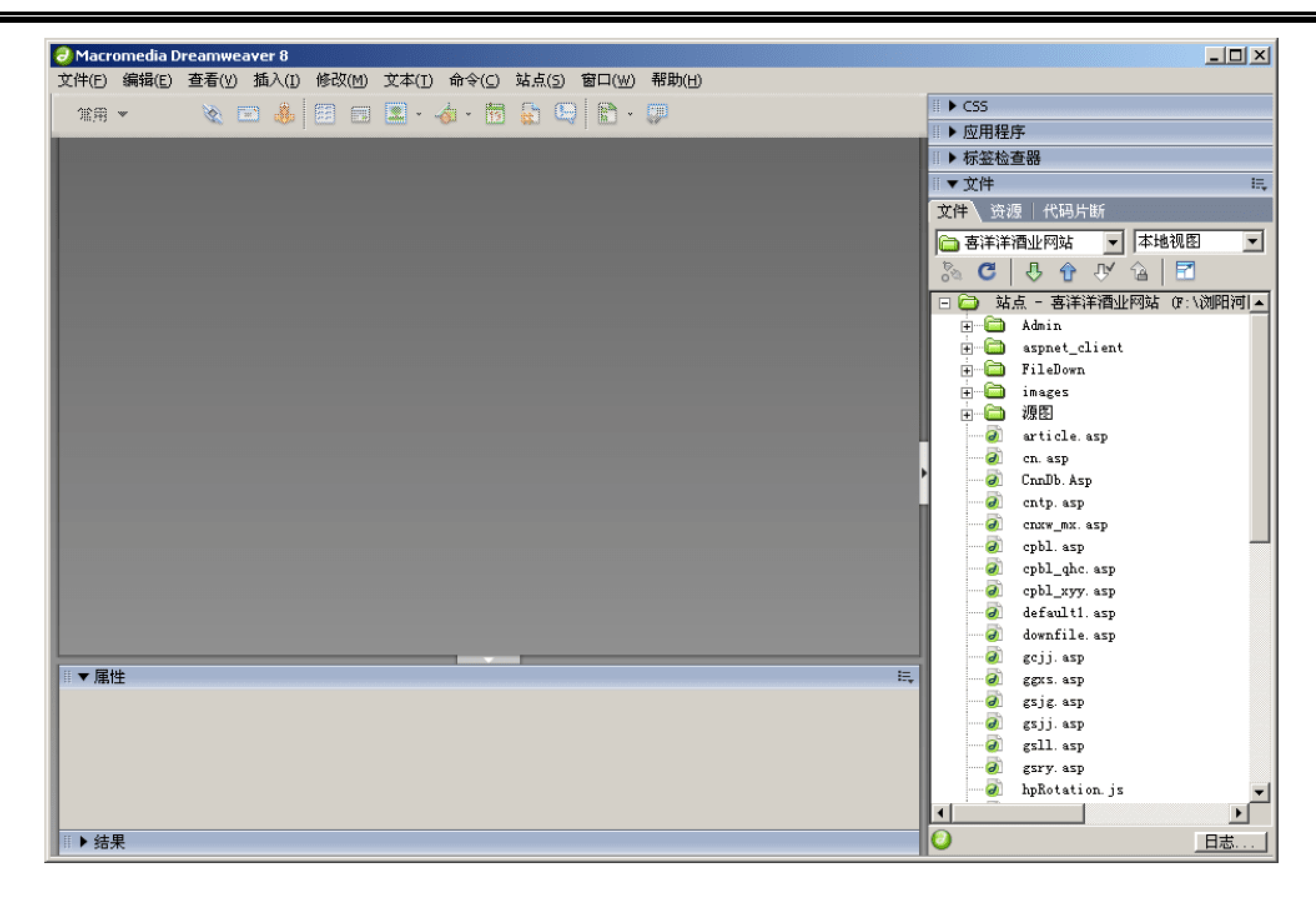


图 2.9　新建站点显示在【文件】面板中

2.3　管 理 站 点

新建站点之后，还可以对已有的站点进行管理，其操作包括编辑站点、复制站点、删除站点、导出站点和导入站点等。

选择【站点】|【管理站点】命令，此时系统将弹出【管理站点】对话框，如图 2.10 所示。

在此对话框中，列出了目前已创建的所有站点，同时提供了用于对站点进行管理的多个按钮。单击相应的按钮，即可对站点执行相应的操作。

图 2.10　【管理站点】对话框

2.3.1　编辑站点

在【管理站点】对话框中，选择要编辑的站点，单击【编辑】按钮，系统将弹出如图 2.11 所示的对话框。

通过“2.2　新建站点”一节的学习，大家也许可以发现用来编辑站点的对话框实际上就是【站点定义】对话框，只是切换到了【高级】选项卡。对初学者来说，可通过站点定义向导来新建站点；而对熟悉 Dreamweaver 的用户，则可在【站点定义】对话框的【高级】选项卡中新建站点。

在【高级】选项卡中，可设置的站点信息包括本地信息、远程信息、测试服务器、遮盖、设计备注、站点地图布局、文件视图列和 Contribute 八类。

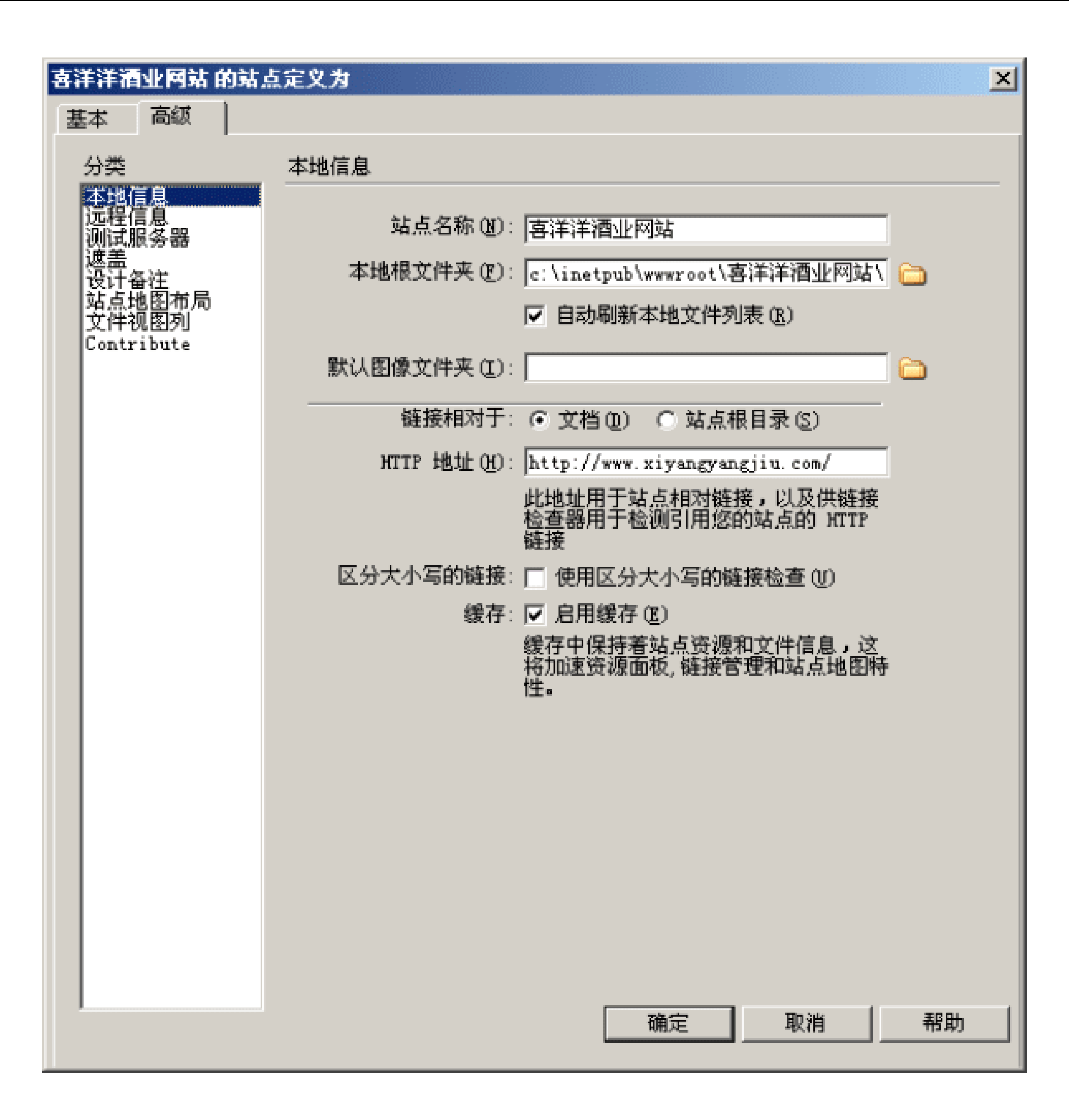


图 2.11　站点定义高级设置对话框

1. 本地信息

【本地信息】类别下的选项主要用于设置新建站点的本地文件夹，各选项如图 2.11 所示。【本地信息】类别中的各项设置含义如下。

- 【站点名称】：新建站点的名称。
- 【本地根文件夹】：在本地磁盘中存储站点文件的文件夹。
- 【自动刷新本地文件列表】：表示每次将文件复制到本地站点时，是否自动刷新本地的文件列表。取消此选项，可提高 Dreamweaver 的速度，但意味着本地的文件列表将不会自动刷新，用户需通过【文件】面板工具栏中的【刷新】按钮来手动刷新文件列表。
- 【默认图像文件夹】：设置站点的默认图像文件夹，该文件夹也是 Dreamweaver 上传用户添加到站点上的图像的默认保存位置。
- 【链接相对于】：设置站点中所创建的链接的相对路径。默认情况下，Dreamweaver 使用文档相对路径创建链接。如果选择站点根目录相对路径，则必须在下面的【HTTP 地址】文本框中指定站点所使用的 URL 地址。
- 【HTTP 地址】：指定站点所使用的 URL 地址。
- 【区分大小写的链接】：用于设置在 Dreamweaver 检查链接时是否区分链接的大小写与文件名的大小写是否匹配。此选项通常用于区分文件名大小写的 UNIX 系统中。
- 【缓存】：用于指定是否创建站点缓存以提高链接和站点管理任务的速度。

2. 远程信息

【远程信息】类别下的选项主要用于设置新建站点的远端文件夹，各选项如图 2.12

所示。

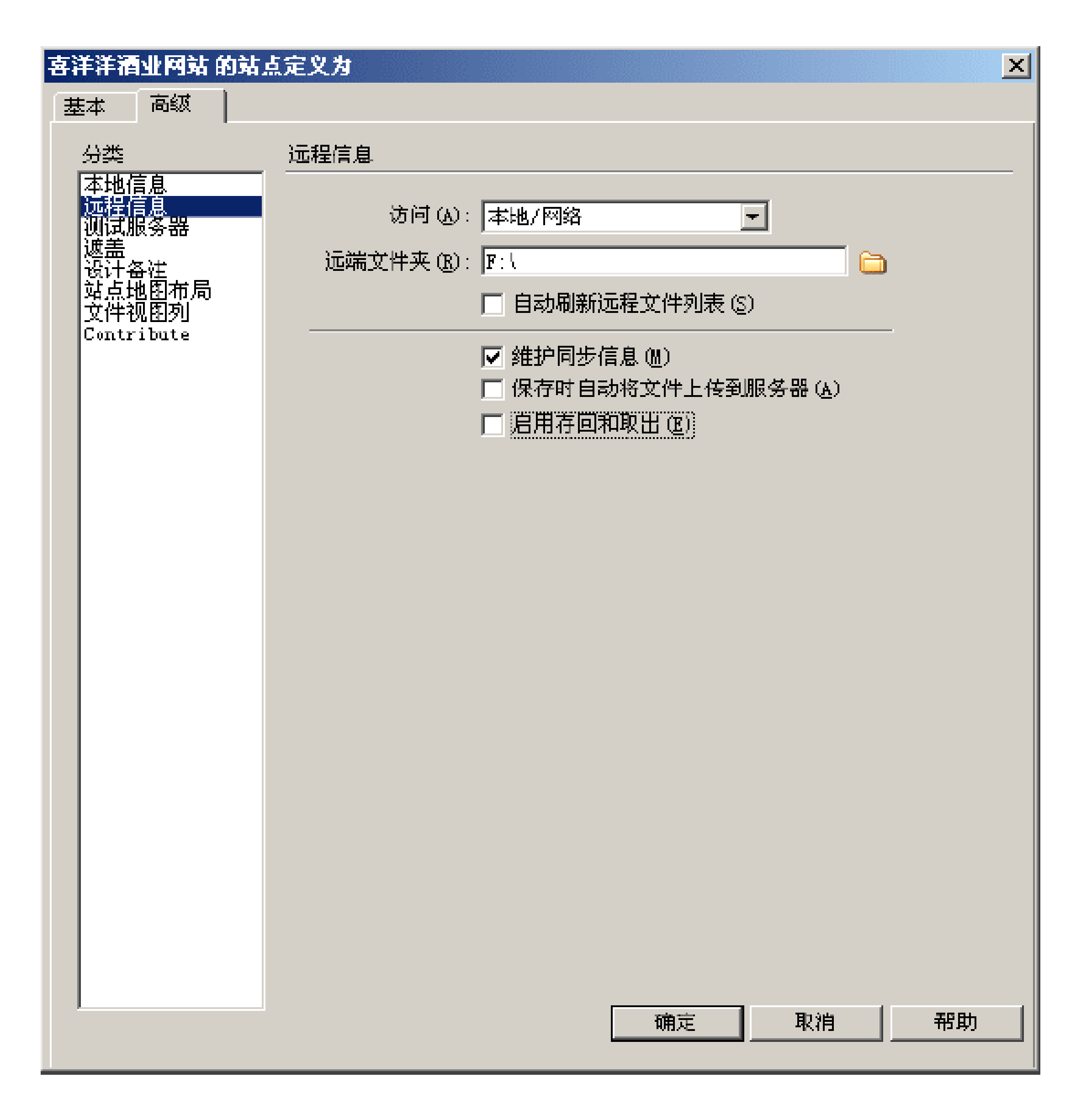


图 2.12　【远程信息】类别下的选项

各选项的含义如下。

- 【访问】：用于设置连接远程服务器的访问方式，其具体选项说明详见 2.2 节中新建站点的第 6 步操作。
- 【远端文件夹】：用于设置在远程服务器上存储站点文件的工作副本的文件夹。
- 【自动刷新远程文件列表】：用于设置在添加和删除文件时是否自动更新【文件】面板中的【远程视图】窗格。
- 【维护同步信息】：用于设置是否自动同步本地和远端文件。
- 【保存时自动将文件上传到服务器】：用于设置是否在保存文件时自动将文件上传到远程站点。
- 【启用存回和取出】：用于设置是否启用存回和取出功能，详见 2.2 节中新建站点的第 7 步操作。

3. 测试服务器

【测试服务器】类别下的选项主要用于指定 Dreamweaver 用来处理动态页面(如 ASP.NET 页面)的测试服务器的相关设置，各选项如图 2.13 所示。

各选项的含义如下。

- 【服务器模型】：设置站点所使用的服务器技术。
- 【访问】：设置访问测试服务器的访问方式，其选项包括【无】、FTP、【本地/网络】和 WebDAV。
- 【测试服务器文件夹】：设置在测试服务器上用来存储站点文件的工作副本的文件夹。
- 【自动刷新测试文件列表】：确认在添加和删除文件时是否自动更新【文件】面板中的【测试服务器】窗格。

- 【URL 前缀】：设置用于测试的 URL 地址前缀。一般来说，URL 前缀不包括任何文件名。

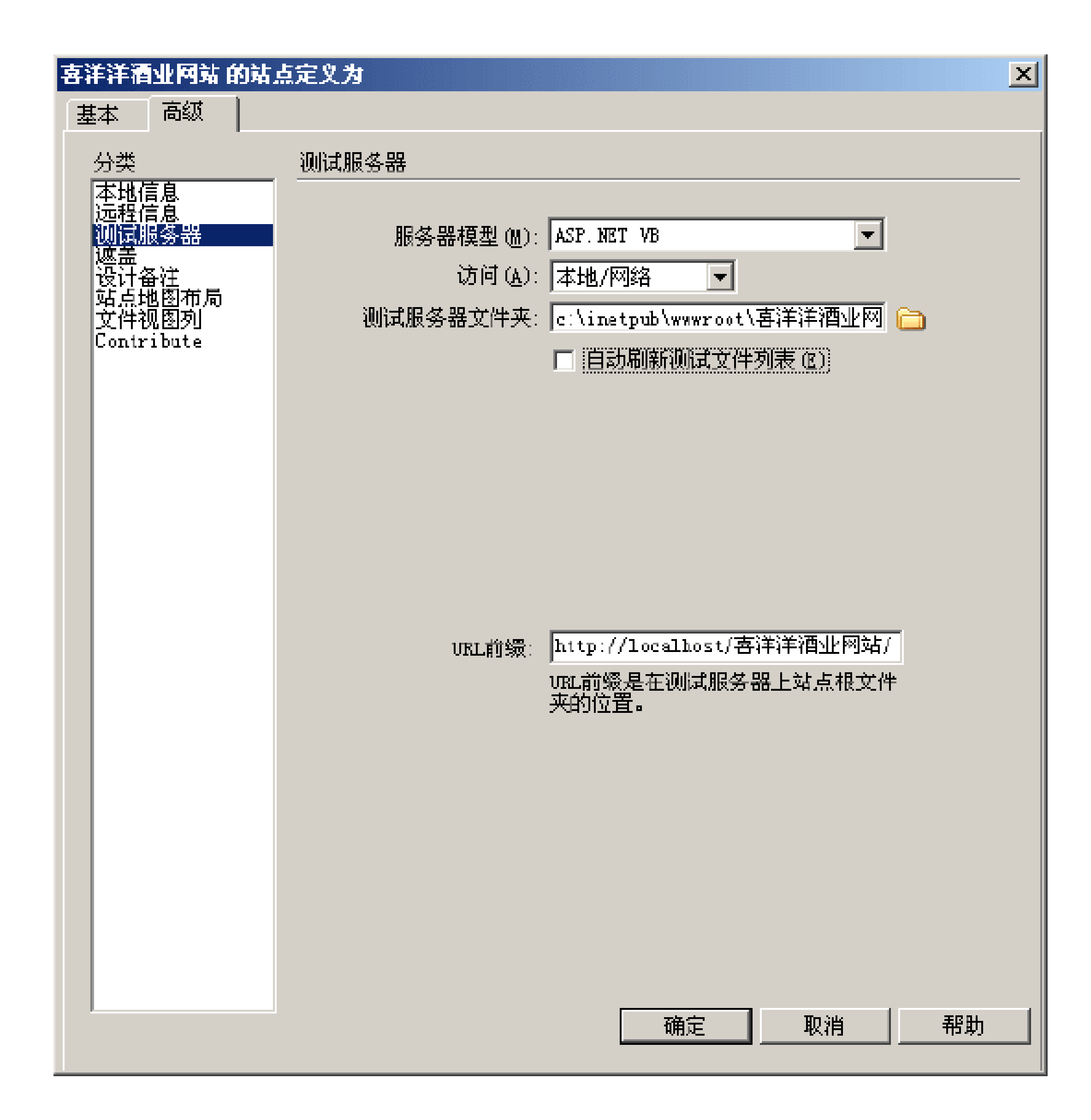


图 2.13　【测试服务器】类别下的选项

4. 遮盖

【遮盖】类别下的选项主要用于设置是否启用遮盖功能以及遮盖特定的文件类型，各选项如图 2.14 所示。

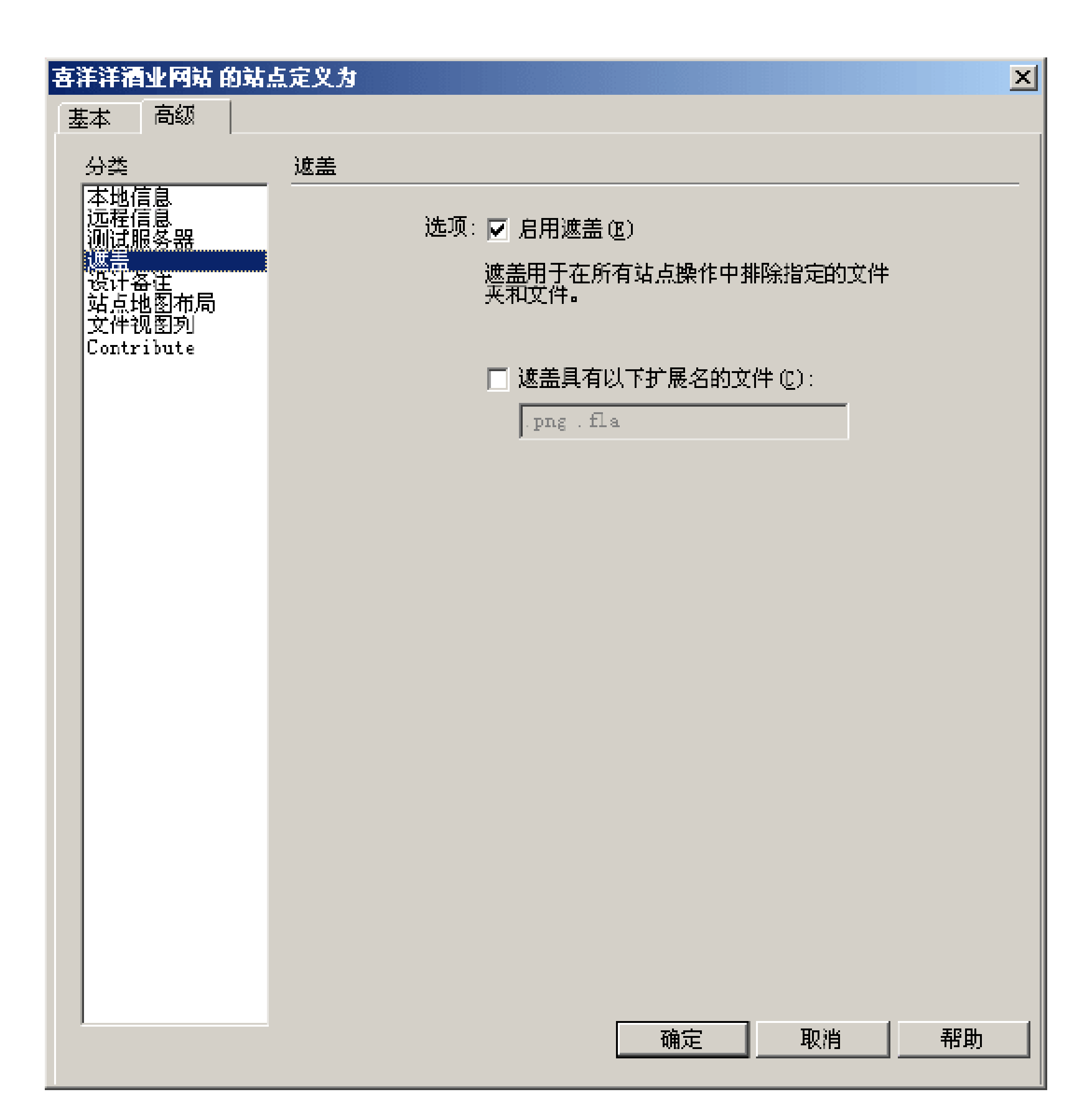


图 2.14　【遮盖】类别下的选项

所谓遮盖，是指在所有站点操作中的排除指定的文件夹和文件。这里的站点操作包括以下几类。

- 上传、获取操作。

- 存回、取出操作。
- 同步操作。
- 使用【资源】面板内容。
- 查找较新的本地文件和查找较新的远端文件。
- 执行站点范围的操作，如检查和更改链接。
- 生成报告。
- 更新模板和库。

通过遮盖功能，可以在以上列出的站点操作中屏蔽指定的文件夹或指定文件类型的文件，但注意不能屏蔽单个文件。

如要遮盖文件夹，可在【文件】面板中选择要进行遮盖的文件夹，然后右击，在弹出的快捷菜单中选择【遮盖】|【遮盖】命令即可。

如要遮盖指定文件类型的文件，可在如图 2.14 所示的【遮盖】类别中选中【遮盖具有以下扩展名的文件】复选框，然后在下面的文本框中输入要遮盖的文件类型的扩展名。如果有多个文件类型，可通过空格进行分隔。

如要取消【遮盖】功能，在【遮盖】类别中取消对【启用遮盖】复选框的选择即可。

5. 设计备注

【设计备注】类别下的选项可帮助用户对设计备注信息进行维护，各选项如图 2.15 所示。

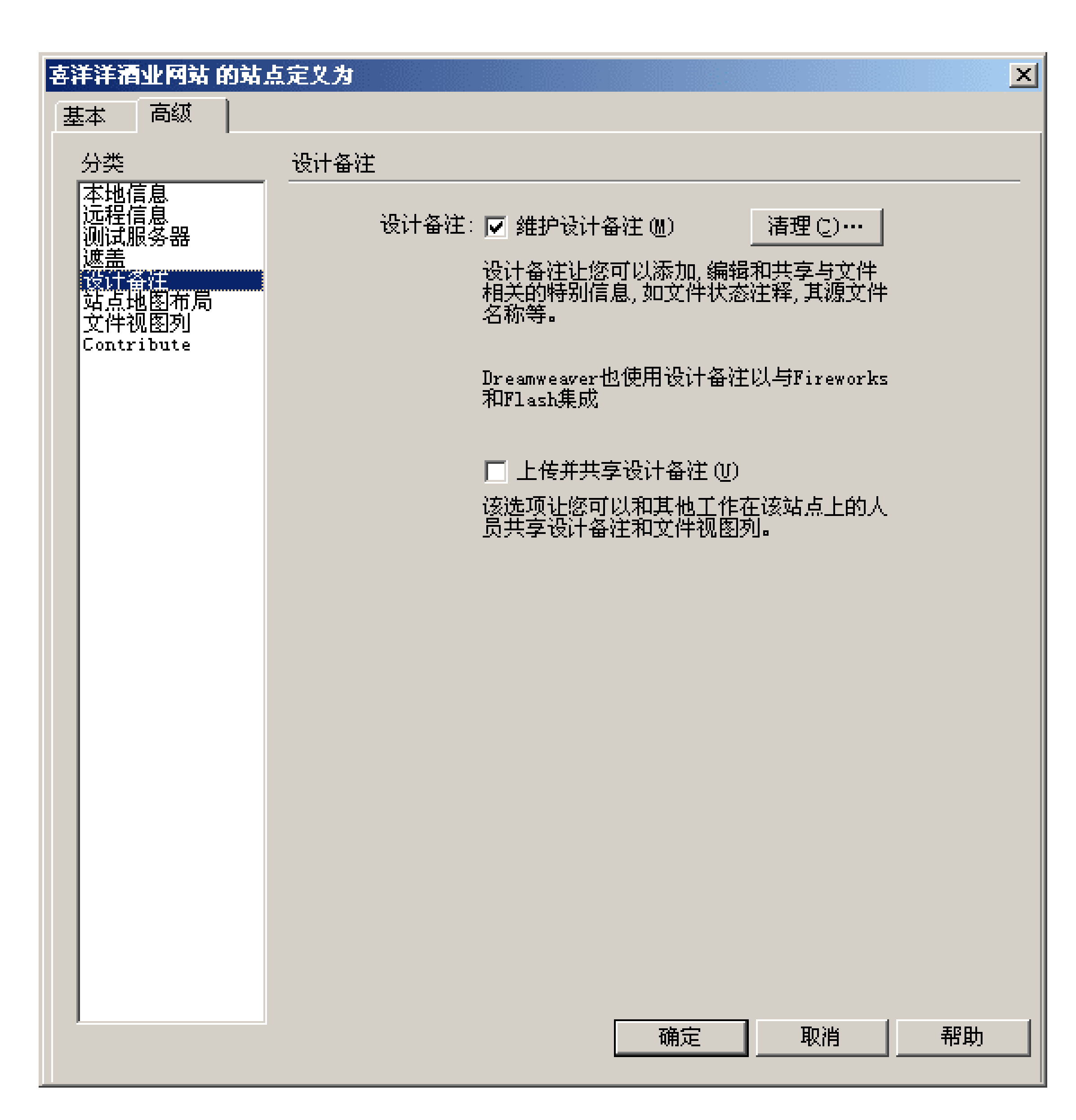


图 2.15　【设计备注】类别下的选项

设计备注是指用户在 Dreamweaver 中为文件所创建的备注信息。设计备注与它们所描述的文件相关联，但存储在单独的文件中。可以在展开的【文件】面板上看到哪些文件有设计备注，通常【设计备注】图标会出现在【备注】列中，如图 2.16 所示。

在图 2.16 中，文件 CnnDb.Asp 所在行的【备注】列显示了【设计备注】图标，表示该

文件存在相关的备注信息。此时，双击该图标，即可查看详细的备注信息，如图 2.17 所示。

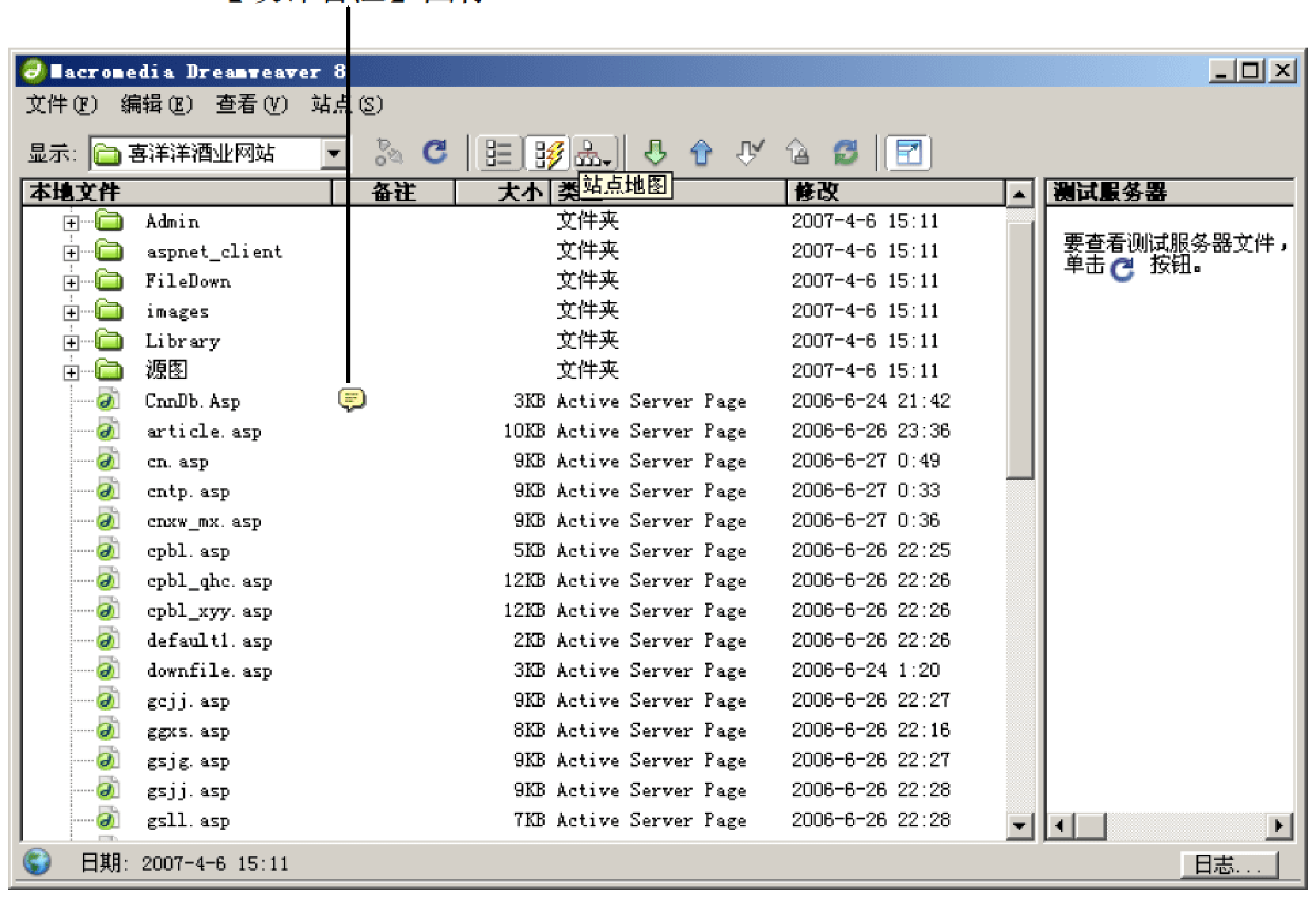


图 2.16　显示【设计备注】

如果要为文件添加备注信息，可在【文件】面板中选择文件，然后右击，在弹出的快捷菜单中选择【设计备注】命令，此时将弹出如图 2.17 所示的【设计备注】对话框。在【备注】文本框中输入备注信息，单击【确定】按钮即可。

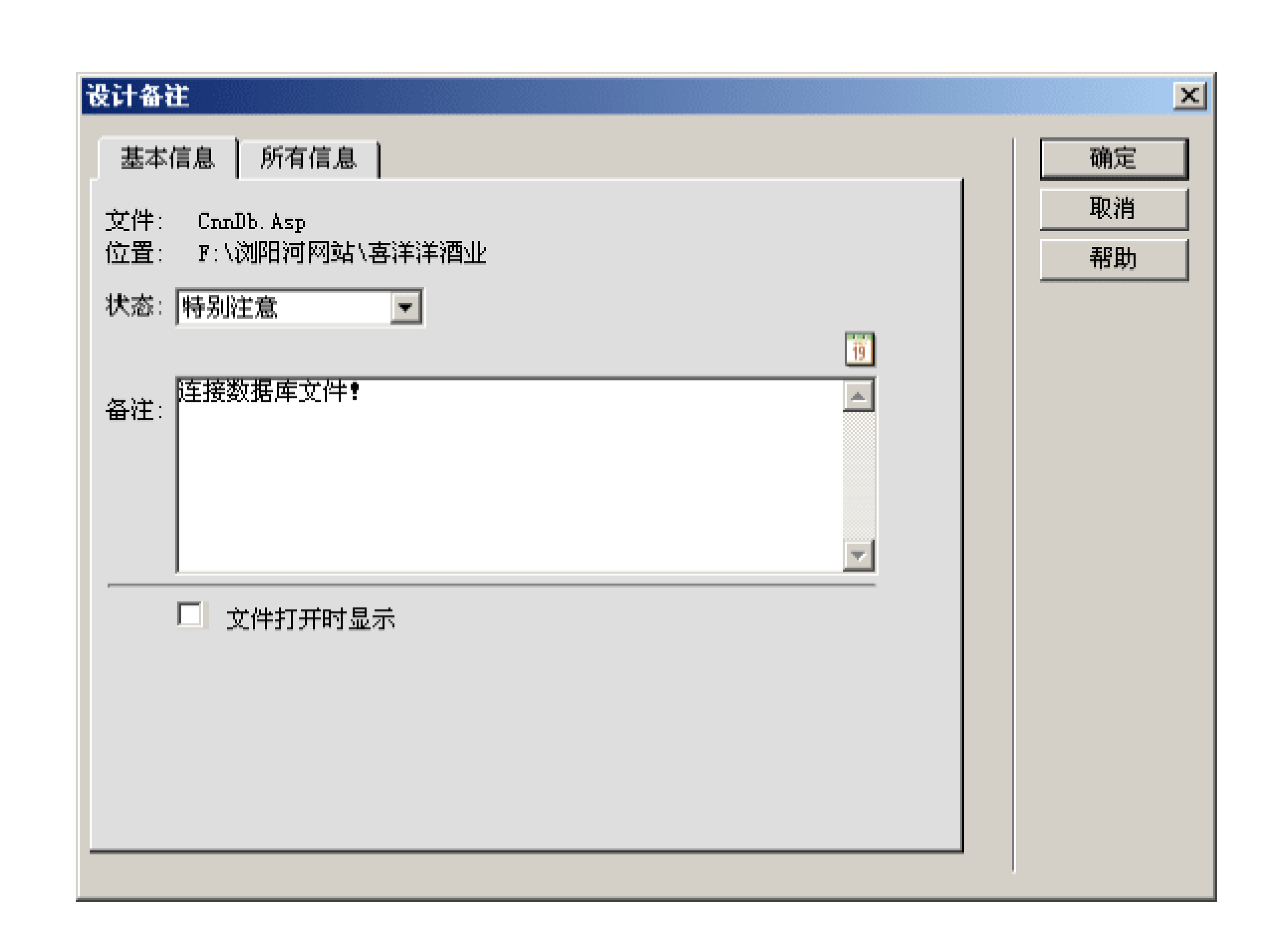


图 2.17　【设计备注】对话框

在如图 2.15 所示的【设计备注】类别中，可对已添加的设计备注信息进行管理。单击【清理】按钮，可删除没有和文件相关联的设计备注信息。选中【上传并共享设计备注】复选框，可将与站点相关的设计备注信息上传至远程服务器中，这样可以与他人一起共享

设计备注。如果取消此选项，则 Dreamweaver 仅在本地文件夹中维护设计备注，而不会将其与站点文件一起上传。

6. 站点地图布局

【站点地图布局】类别下的选项可对站点地图的外观进行自定义设置，各选项如图 2.18 所示。

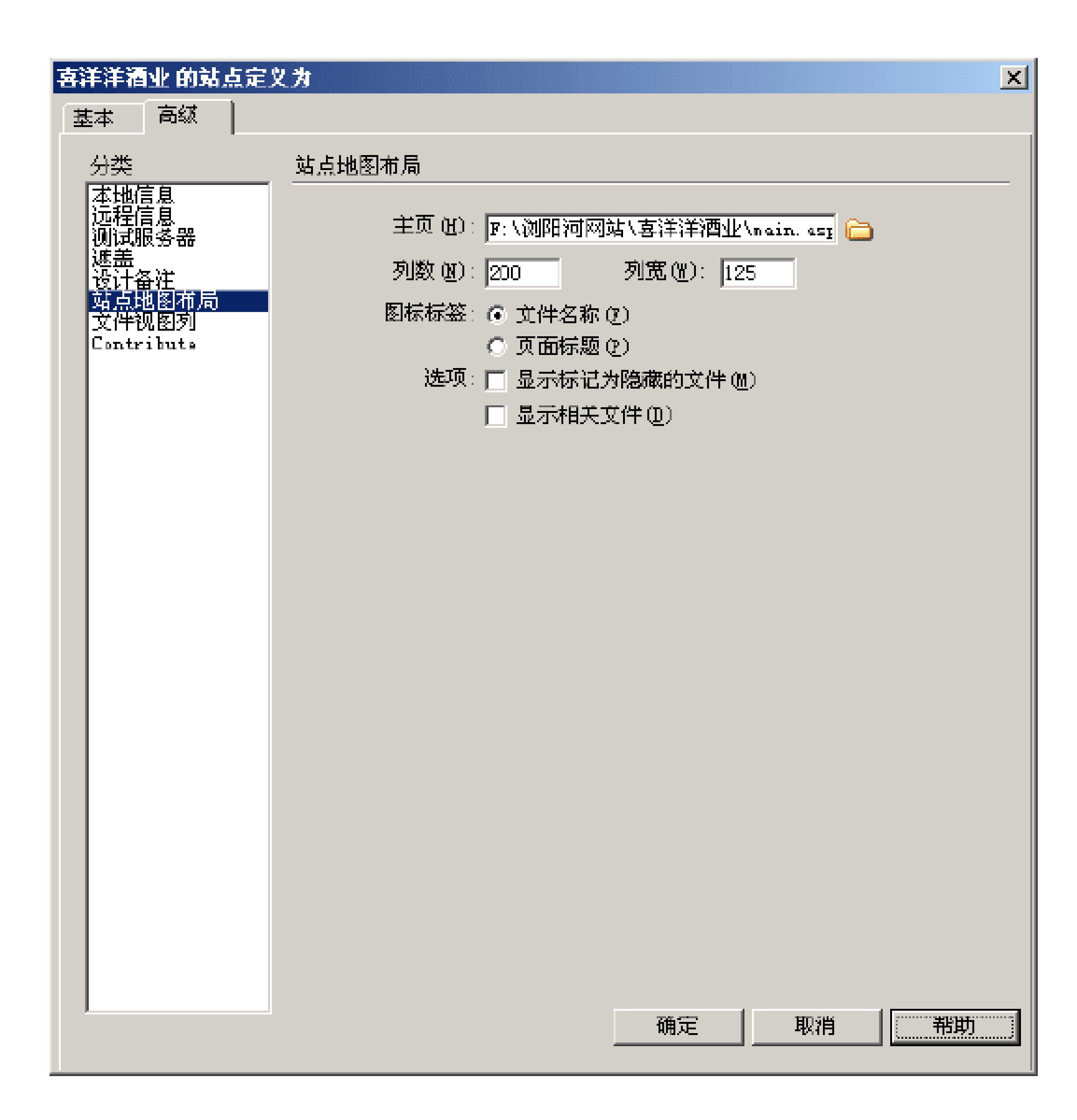


图 2.18　【站点地图布局】类别下的选项

站点地图实际上就是一个站点结构的示意图，它从主页开始显示两个级别深度的站点结构。站点地图将页面显示为图标，并按在源代码中出现的顺序来显示链接，如图 2.19 所示。

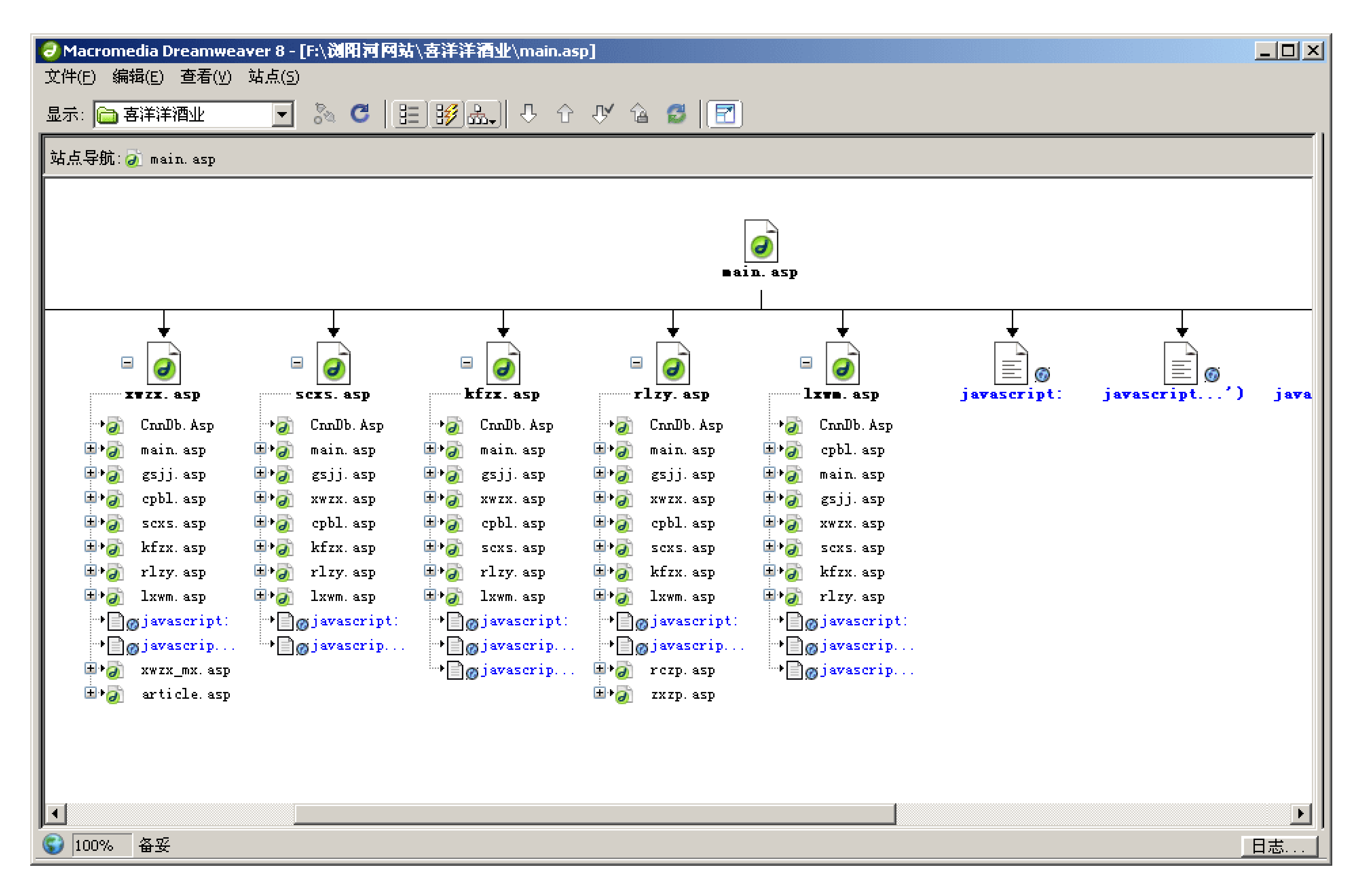


图 2.19　站点地图

各选项的含义如下。

- 【主页】：用于确定站点的主页，其文件必须是在本地站点上，这是站点地图所必须设置的。如果没有设置站点主页或所设置的主页文件无法找到，则当打开站点地图时，系统将提示选择主页。
- 【列数】：设置站点地图窗口中每行所显示的列数。
- 【列宽】：设置站点地图中每列的宽度(以像素为单位)。
- 【图标标签】：设置在站点地图中与文档图标一起显示的名称是文件名称还是页面标题。
- 【显示标记为隐藏的文件】：确认在站点地图中是否显示用户标记为隐藏的页面文件。
- 【显示相关文件】：确认在站点地图中是否显示该站点层次结构中的所有相关文件。

7. 文件视图列

【文件视图列】类别下的选项主要用于对【文件】面板中所显示的文件视图列进行设置，各选项如图 2.20 所示。

图 2.20　【文件视图列】类别下的选项

在此对话框中，可以对文件视图列进行添加和删除操作，以及设置列的显示顺序及相关属性。

如要添加列，可单击 + 按钮，在下面的【列名称】文本框中输入该列的显示名称，分别在【与设计备注关联】和【对齐】下拉列表框中选择与设计备注关联的值以及对齐方式，单击【确定】按钮即可。

提示： 只有在【与设计备注关联】下拉列表框中设置相应的值，该列在【文件】面板中才会显示数据。

如要删除列，可在上方列表中选择要删除的列，单击[-]按钮即可。这里仅能删除用户新添加的列。对于 Dreamweaver 中内置的文件视图列，如果不想让它显示，可取消该列的【显示】选项。

通过单击▲按钮和▼按钮，可更改选定列的显示顺序。但【名称】列的顺序无法更改，它始终显示为第一列。

8. Contribute

Contribute 类别下的选项主要用于设置是否启用 Contribute 兼容性，其选项如图 2.21 所示。

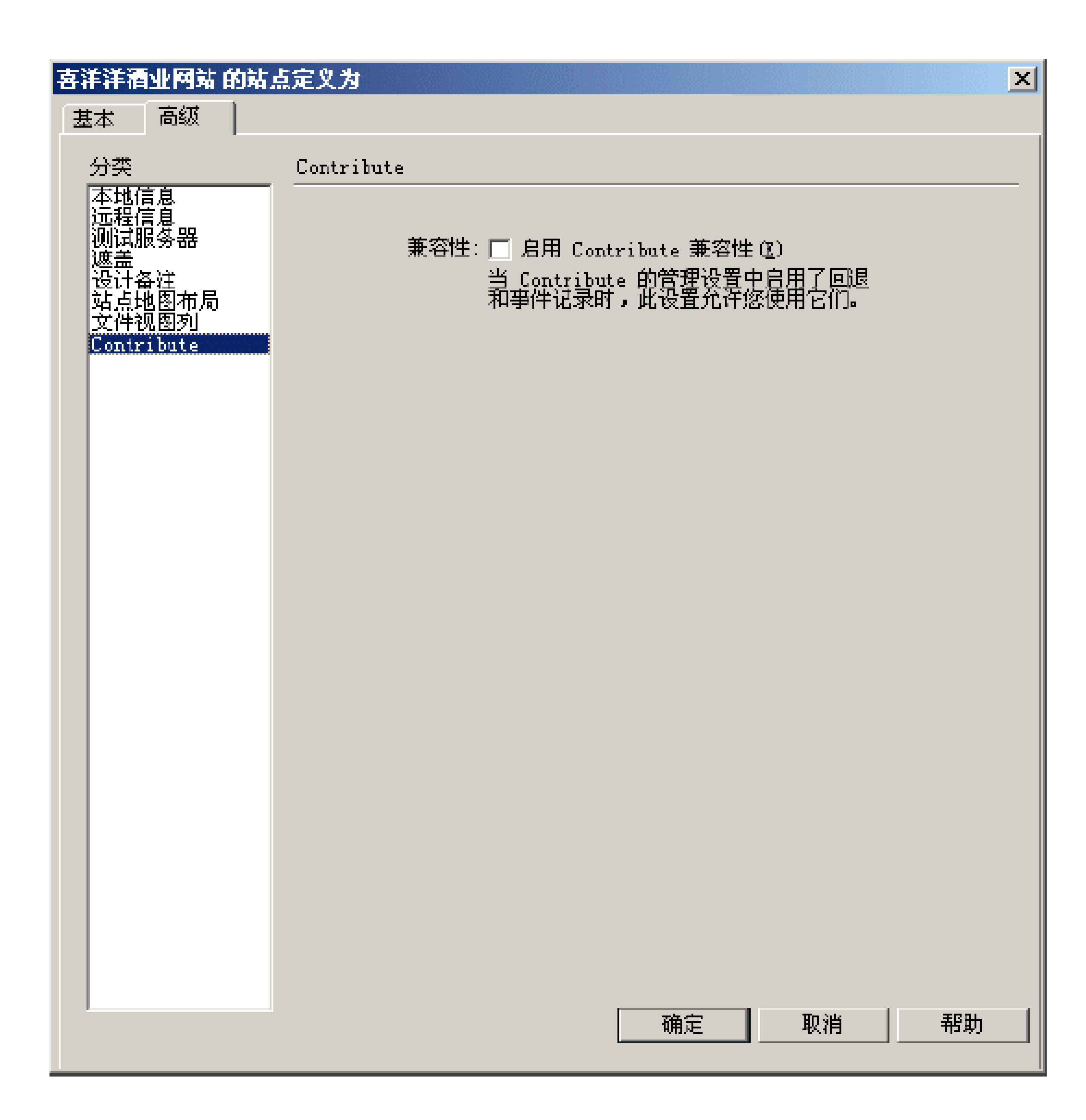


图 2.21　Contribute 类别下的选项

Contribute 是一个站点管理工具，它结合了 Web 浏览器和基本的 Web 页编辑器的功能，可对站点进行管理。Contribute 拥有一个网站的所有复杂的程序代码，同时它还提供了一个类似于 Word 处理器的工具，帮助用户处理页面中的文本及表格。

如果启用 Contribute 兼容性，则可通过 Dreamweaver 启动 Contribute 来执行站点管理任务。但是，必须保证 Dreamweaver 与 Contribute 安装在同一台计算机上。

2.3.2　复制站点

如果希望创建多个结构相同或类似的站点，可利用站点的复制功能。首先创建一个基准站点，并以该站点为基础复制出多个站点，然后根据需要分别对各站点进行编辑，这样可大大提高工作效率。

复制站点的操作很简单，在如图 2.10 所示的【管理站点】对话框中，选择要复制的站点，单击【复制】按钮，即可复制出一个新站点。新站点将显示在【管理站点】对话框的站点列表中，其名称在原站点名称的基础上添加了“复制”两个字，如图 2.22 所示。

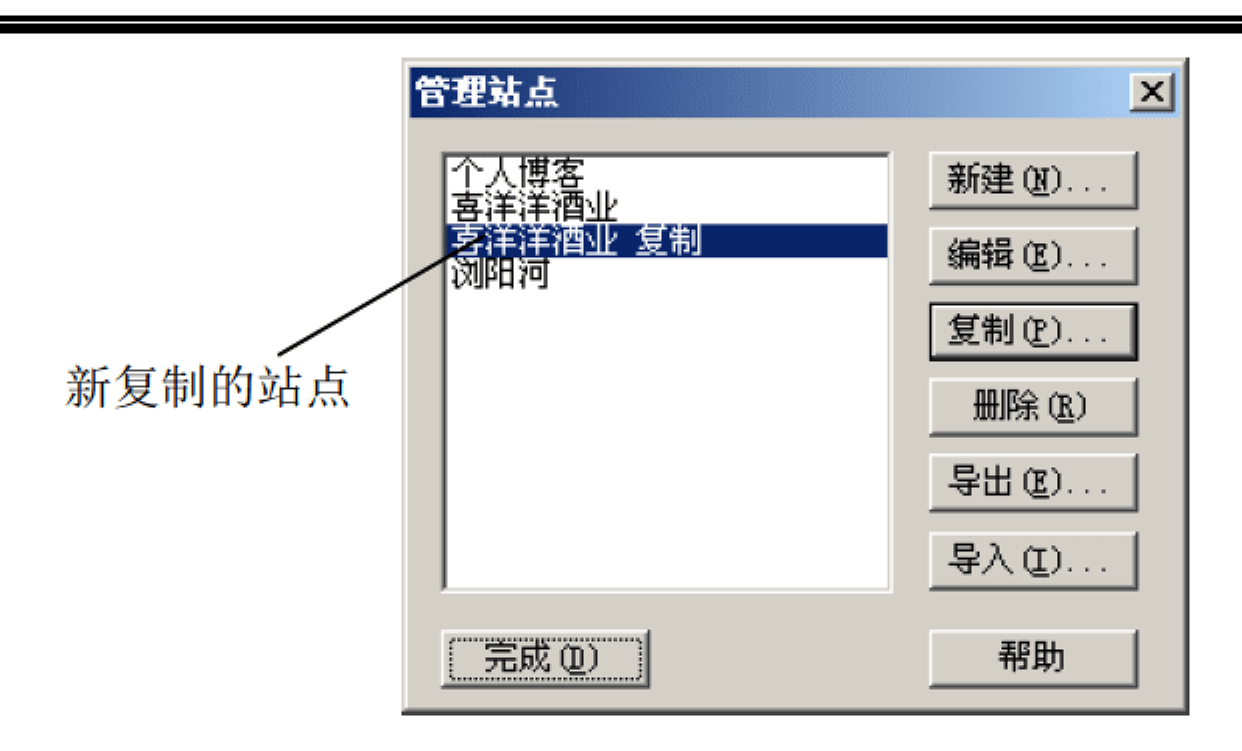


图 2.22　复制站点

2.3.3　删除站点

如果不再需要利用 Dreamweaver 对某个站点进行操作，可将其从站点列表中删除，其操作如下。

在如图 2.10 所示的【管理站点】对话框中，选择要删除的站点，单击【删除】按钮，即可将其从站点列表中删除。

提示： 删除操作实际上只是删除了 Dreamweaver 与该站点之间的关系，而不会真正删除站点文件。实际的站点文件仍保留在原来的磁盘位置上，可以重新创建指向其位置的新站点，对其进行重新管理。

2.3.4　导出站点

为了做好站点的备份工作，用户可以将指定的站点导出为一个站点定义文件。注意，这里导出的仅仅是站点的相关设置信息，不包括站点文件。

要执行站点导出功能，可在如图 2.10 所示的【管理站点】对话框中，选择所要导出的站点，单击【导出】按钮，在弹出的【导出站点】对话框中选择要保存的位置，输入文件名，单击【保存】按钮即可。

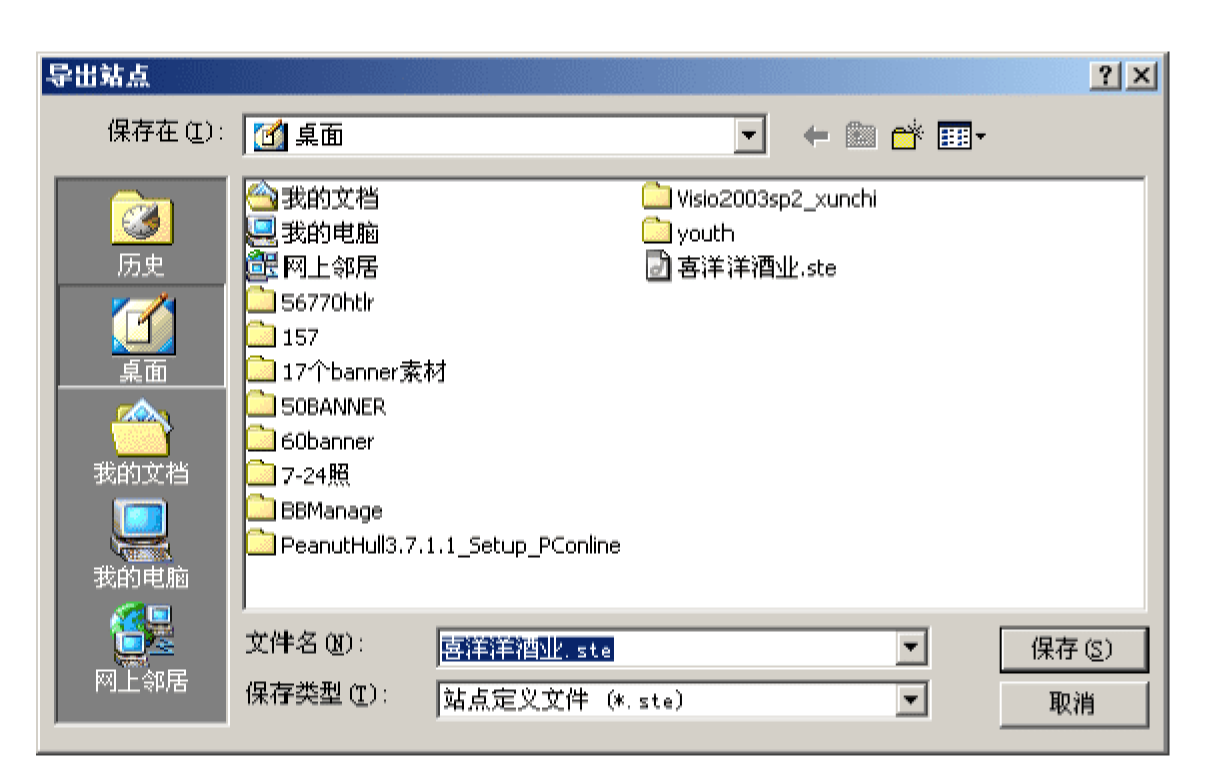


图 2.23　【导出站点】对话框

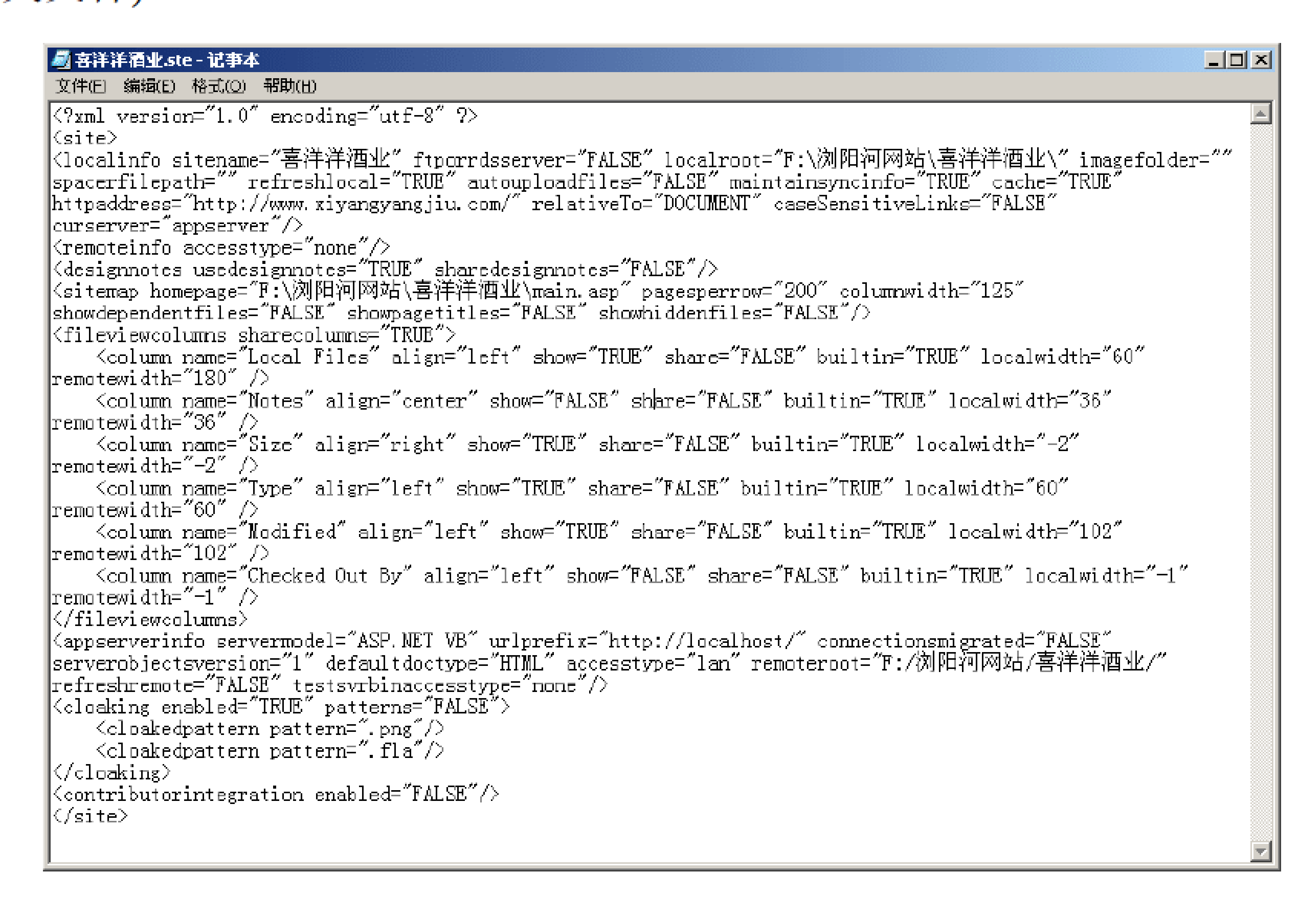
导出的站点文件为以.ste 为扩展名的 XML 文件，可通过“记事本”将其打开，如图 2.24 所示。

由此可见，站点定义文件中所包含的仅仅是该站点的设置信息。

此外，还可以同时导出多个站点。其具体操作如下。

(1) 选取多个站点。对于多个站点的选择有两种方式：对于多个间隔的站点，可按住 Ctrl 键单击每个站点；对于某一范围内的所有站点，可按住 Shift 键单击该范围的第一个站点和最后一个站点。

(2) 单击【导出】按钮，在依次弹出的【导出站点】对话框中确认保存路径和文件名，单击【保存】按钮即可。此时，Dreamweaver 会为导出的每一个站点生成一个单独的.ste 文件(站点定义文件)。




```
<?xml version="1.0" encoding="utf-8" ?>
<site>
<localinfo sitename="喜洋洋酒业" ftporrdsserver="FALSE" localroot="F:\浏阳河网站\喜洋洋酒业\" imagefolder=""
spacerfilepath="" refreshlocal="TRUE" autouploadfiles="FALSE" maintainsyncinfo="TRUE" cache="TRUE"
httpaddress="http://www.xiyangyangjiu.com/" relativeTo="DOCUMENT" caseSensitiveLinks="FALSE"
curserver="appserver"/>
<remoteinfo accesstype="none"/>
<designnotes usedesignnotes="TRUE" sharedesignnotes="FALSE"/>
<sitemap homepage="F:\浏阳河网站\喜洋洋酒业\main.asp" pagesperrow="200" columnwidth="125"
showdependentfiles="FALSE" showpagetitles="FALSE" showhiddenfiles="FALSE"/>
<fileviewcolumns sharecolumns="TRUE">
    <column name="Local Files" align="left" show="TRUE" share="FALSE" builtin="TRUE" localwidth="60"
remotewidth="180" />
    <column name="Notes" align="center" show="FALSE" share="FALSE" builtin="TRUE" localwidth="36"
remotewidth="36" />
    <column name="Size" align="right" show="TRUE" share="FALSE" builtin="TRUE" localwidth="-2"
remotewidth="-2" />
    <column name="Type" align="left" show="TRUE" share="FALSE" builtin="TRUE" localwidth="60"
remotewidth="60" />
    <column name="Modified" align="left" show="TRUE" share="FALSE" builtin="TRUE" localwidth="102"
remotewidth="102" />
    <column name="Checked Out By" align="left" show="FALSE" share="FALSE" builtin="TRUE" localwidth="-1"
remotewidth="-1" />
</fileviewcolumns>
<appserverinfo servermodel="ASP.NET VB" urlprefix="http://localhost/" connectionsmigrated="FALSE"
serverobjectsversion="1" defaultdoctype="HTML" accesstype="lan" remoteroot="F:/浏阳河网站/喜洋洋酒业/"
refreshremote="FALSE" testsvrbinaccesstype="none"/>
<cloaking enabled="TRUE" patterns="FALSE">
    <cloakedpattern pattern=".png"/>
    <cloakedpattern pattern=".fla"/>
</cloaking>
<contributorintegration enabled="FALSE"/>
</site>
```

图 2.24　导出的站点定义文件

2.3.5　导入站点

与导出站点相反，导入站点是将指定的.ste 文件(站点定义文件)导入至 Dreamweaver 中，生成一个新的站点。

要执行站点导入功能，可在如图 2.10 所示的【管理站点】对话框中单击【导入】按钮，在弹出的【导入站点】对话框中选择要导入的.ste 文件，单击【打开】按钮即可。

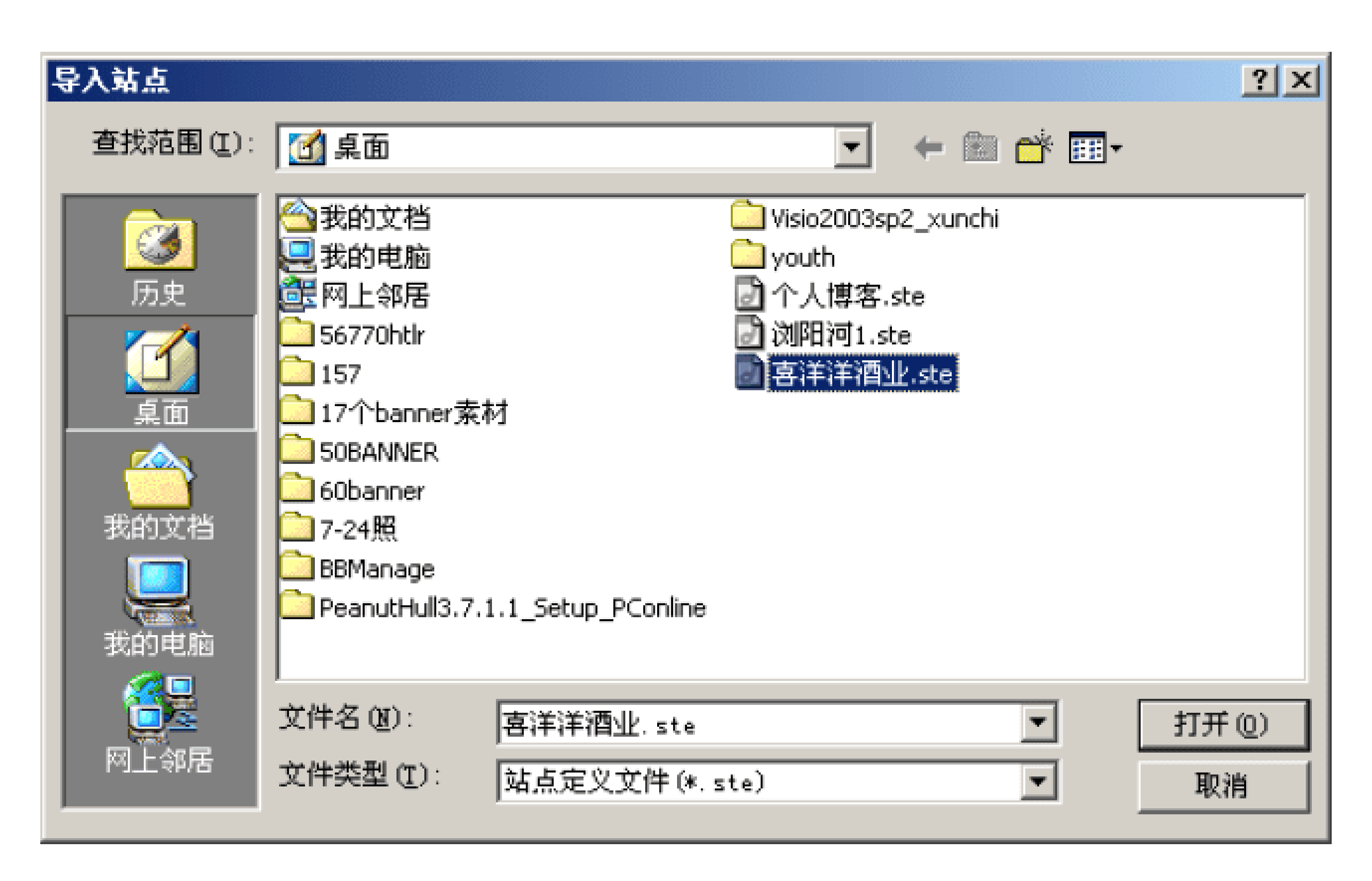


图 2.25　【导入站点】对话框

导入站点后，将在 Dreamweaver 中生成一个新的站点，该站点名称与导出该站点定义文件时对应的站点的名称一致。如果 Dreamweaver 中已存在该名称的站点，系统将会提示已存在同名站点，并将为导入的站点重命名，如图 2.26 所示。

图 2.26　提示站点重名

与导出站点一样，可以同时导入多个站点。此时，只需在【导入站点】对话框中，同时选取多个.ste 文件，单击【打开】按钮即可。

2.4　站点的高级操作

前面介绍了在 Dreamweaver 中对站点的一些基本操作，下面将进一步介绍站点的部分高级操作。

2.4.1　管理站点资源

所谓站点资源，是指站点中所存储的各种页面元素，包括图像、影片、Flash、声音、颜色、链接、脚本等。这些资源也是各个站点不可或缺的。随着站点的一步步开发，其中所积累的资源将会越来越多。此时，对资源进行集中管理，已是一件非常必要的事情。在 Dreamweaver 8 中，提供了对站点资源的管理功能。通过管理站点资源，可以轻松地跟踪和预览已存储在站点中的所有资源，也可以将其中某种资源直接拖至当前打开的文档中；另外，如果需要在多个站点中使用同一资源，也可将其复制至其他站点中使用。

1. 查看资源

可以通过【资源】面板来查看和管理当前站点中的资源。【资源】面板中显示了与【文档】窗口中的活动文档相关联的所有站点资源，如图 2.27 所示。

所有资源是通过分类显示的，列表中仅能显示一个类别的资源。在【资源】面板的左侧，显示了资源的相关分类。从图 2.27 中可以看出，在 Dreamweaver 8 中，资源被分为图像、颜色、URLs(链接)、Flash、Shockwave、影片、脚本、模板和库等九大类，其中模板和库是两种特殊类型的资源(这点，将会在后面进行详细介绍)。默认情况下，【图像】类别处于选定状态。通过单击不同的类别按钮，可查看该类别的相关资源。

在【资源】面板中，提供了【站点】和【收藏】两种视图。【站点】视图显示的是站点的所有资源，而【收藏】视图仅显示用户明确选择并添加至收藏中的资源。

提示：【站点】视图和【收藏】视图对【模板】类别和【库】类别无效。当查看【模板】或【库】类别资源时，【站点】和【收藏】两个单选按钮不可用。

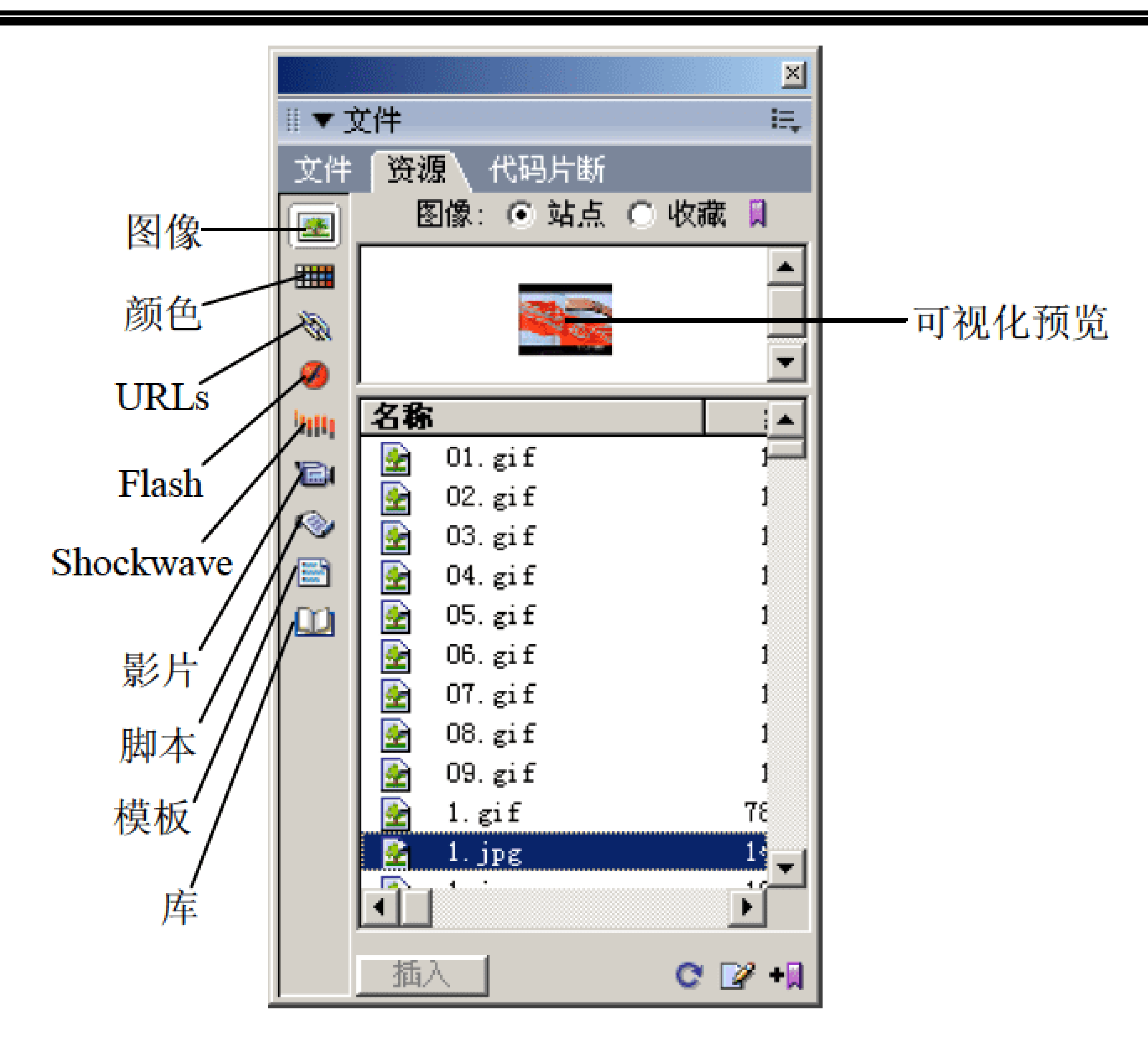


图 2.27　查看资源

在默认情况下，所显示的指定类别的资源是以名称字母的升序进行排列。通过单击不同的列标题，可改变资源的排序方式。通过拖动两个列标题之间的分隔线，可改变前一列的宽度。

当选择某一资源，在面板顶部的预览区域将会显示该资源的可视化预览。

如果在站点中添加或删除资源时，其所做的修改不会立即出现在【资源】面板中，此时可通过单击【资源】面板下方的 按钮来刷新显示。但是，如果对资源的添加和删除操作是在 Dreamweaver 外部进行的(如通过 FrontPage 或 Visual Web Developer 2005 等软件添加或删除站点中的资源)，则必须重新生成站点缓存才能更新【资源】面板。

2. 使用资源

资源的作用在于使用，那么如何使用【资源】面板中所列的资源呢？一般来说，可以将大多数类型的资源插入到当前打开的文档中，而模板类除外(模板仅能应用于整个文档，而不能将其插入到文档中)。对于颜色和 URL，还可将其应用于【设计】视图中所选定的文本。

将资源插入到文档中的方法有两种。

- 在当前打开文档的【设计】视图中，将插入点放置在资源要插入的位置，然后在【资源】面板中选择所要插入的资源，并将其拖动至文档的插入点。
- 在当前打开文档的【设计】视图中，将插入点放置在资源要插入的位置，然后在【资源】面板中选择所要插入的资源，单击【资源】面板下方的【插入】按钮(如果是【颜色】类别的资源，则单击【应用】按钮)即可。

如果要将【颜色】或 URL 资源应用于选定的文本，其方法类似，只是首先并不是放置插入点，而是选择要应用资源的文本，然后选择资源并拖动或单击【插入】按钮(对于【颜色】资源，单击【应用】按钮)即可。

3. 编辑资源

一般来说，对于除脚本之外的所有资源，均可对其进行编辑。其类别不同，编辑的方式也不一样。

对于图像、Flash、Shockwave、影片等类别的资源，通过双击指定资源或选取资源并单击按钮，可对其执行编辑操作。此时，Dreamweaver 将启用与该资源类型相关联的外部编辑应用程序来对其进行编辑。如果没有与该资源类型相关联的外部应用程序，用户可在【首选参数】对话框中自行设置，具体方法如下：

(1) 选择【编辑】|【首选参数】命令，在弹出的【首选参数】对话框中，选择【文件类型/编辑器】类别，如图 2.28 所示。

图 2.28　设置外部编辑器

(2) 在【扩展名】列表中，选择要更改外部编辑器的文件类型，单击【外部代码编辑器】文本框右边的【浏览】按钮，在弹出的【选择外部编辑器】对话框中选择相应的应用程序即可。

一个文件类型可同时设置多个外部编辑器。通过单击【设为主要】按钮，可选定其中的一个作为首选编辑器。

对于颜色和 URL 类型的资源，只能在【收藏】视图中编辑。

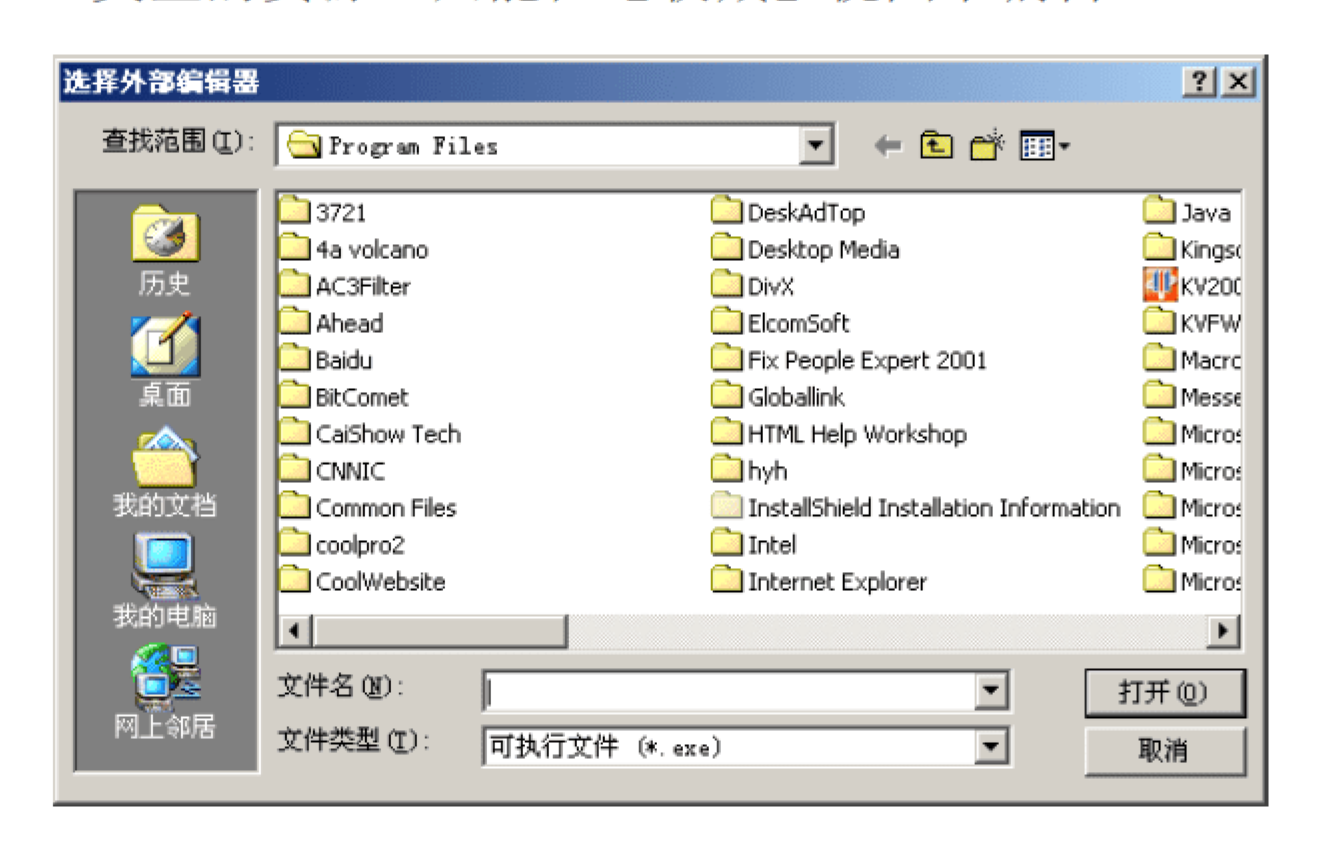


图 2.29　【选择外部编辑器】对话框

如果需要编辑颜色类资源，可在【收藏】视图中选择所要编辑的颜色，然后双击或单击按钮，此时将弹出颜色选择器，如图 2.30 所示。

在颜色选择器中选取更改后的颜色，即可完成对指定颜色资源的编辑。

如果需要编辑 URL 类资源，可在【收藏】视图中选择所要编辑的 URL 资源，然后双击或单击按钮，此时将弹出【编辑 URL】对话框，如图 2.31 所示。

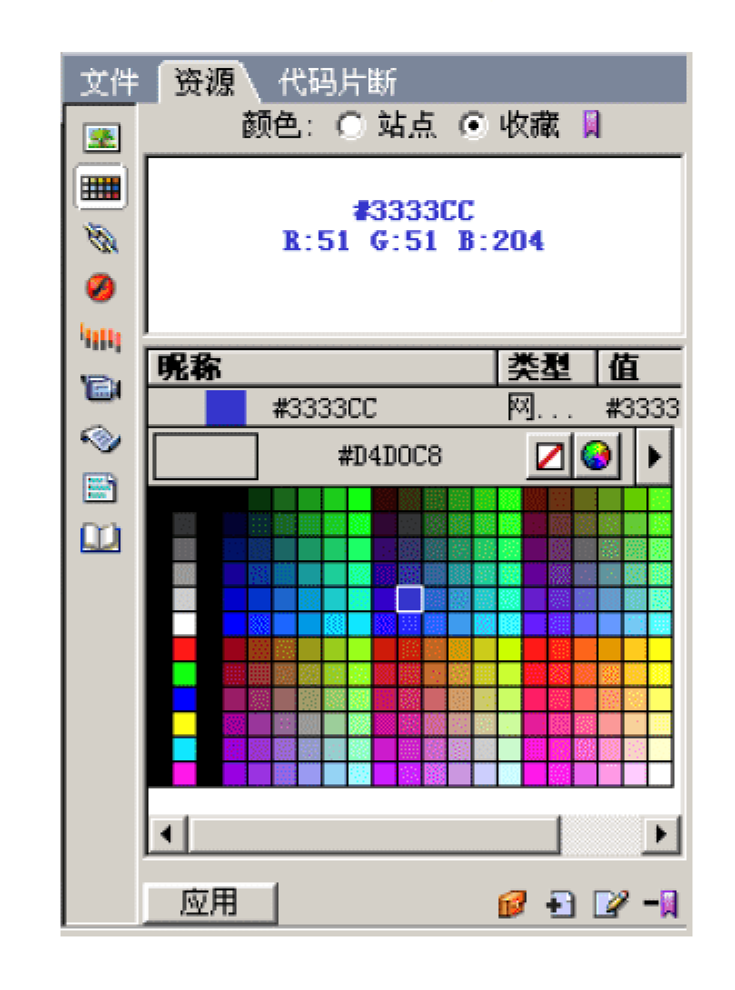


图 2.30　编辑颜色类资源

图 2.31　编辑 URL 类资源

在 URL 文本框中，输入要更改的值，单击【确定】按钮即可。

4. 模板和库

模板和库是 Dreamweaver 中两种特殊类型的资源，它们是一种链接资源，对其进行编辑将更新已应用这些资源的相关文档。

模板是一种特殊类型的文档，用于设置相对固定的页面布局。可以基于模板创建文档，从而使创建的文档继承模板的页面布局。同时，在设计模板时，可指定在基于模板的文档中用户可以编辑文档的哪些区域。通过模板，可在多个页面上重复使用同一页面布局。而其最大的好处就在于，当更新模板时将同时更改应用该模板的所有页面上的布局。

库是一种以.lbi 为扩展名的特殊的 Dreamweaver 文件，它用于存储要在整个站点中经常重复使用或更新的资源(如站点的版权信息或徽标)，这些资源包括图像、表格、声音、Flash 文件等。库里存储的资源又称为库项目。每当更改库项目的内容时，将同步更新所有使用该项目的页面。事实上，在使用库项目时，Dreamweaver 并不是在页面中插入库项目，而是插入一个指向库项目的链接，如图 2.32 所示。

从图 2.32 中可以看出，Dreamweaver 向页面中插入了该项目的 HTML 源代码副本，同时添加了一个包含对原始外部项目的引用的 HTML 注释。与模板相比，库主要用于个别的设计元素，而模板则用于控制更大的设计区域。

对于模板和库的各项具体操作，这里不再详细叙述，读者可参考相关的帮助文档。

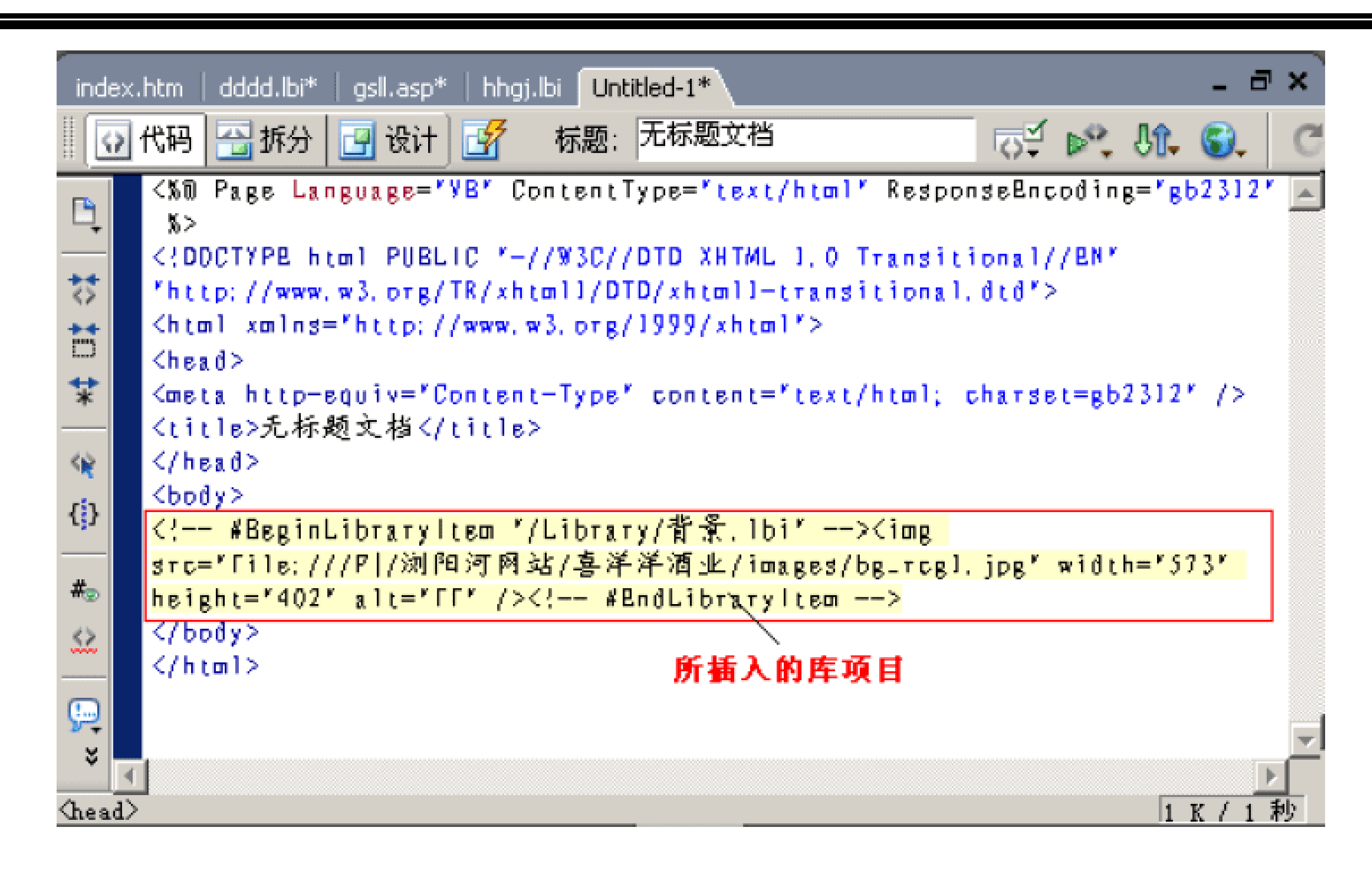


图 2.32　插入库项目

2.4.2　同步站点文件

如果站点设置了远程服务器，那么在创建了本地站点和远程站点之后，可以在这两者之间实现文件同步。为确保文件安全，可在同步站点之前验证要将哪些文件上传到远程服务器上或从远程服务器上获取哪些文件。此外，在同步文件之后，还可确认对哪些文件进行更新。

如要同步站点文件，其操作如下。

(1)　选择【站点】|【同步站点范围】命令，系统将弹出【同步文件】对话框，如图 2.33 所示。

图 2.33　【同步文件】对话框

在【同步】下拉列表框中包括两个选项：【整个‘喜洋洋酒业’站点】(注意，喜洋洋酒业为站点名称)和【仅选中的远端文件】。前者是同步整个站点的所有文件，而后者是同步在【文件】面板的【远程】视图中所选择的文件。在【方向】下拉列表框中提供了 3 个选项：【放置较新的文件到远程】、【从远程获得较新的文件】和【获得和放置较新的文件】。

- 【放置较新的文件到远程】：将上传远端服务器上不存在的或自从上次上传以来已更改的所有本地文件。
- 【从远程获得较新的文件】：将获取本地不存在的或自从上次获取以来已更改的所有远端文件。
- 【获得和放置较新的文件】：将所有文件的最新版本更新至本地和远端站点上。

(2)　单击【预览】按钮，此时 Dreamweaver 将根据用户在【同步】和【方向】下拉列

表框中的选择，检测本地和远端站点上文件的较新版本，如图 2.34 所示。在检测完成后，Dreamweaver 将在【同步】对话框中显示检测的结果，如图 2.35 所示。

图 2.34　检测较新文件

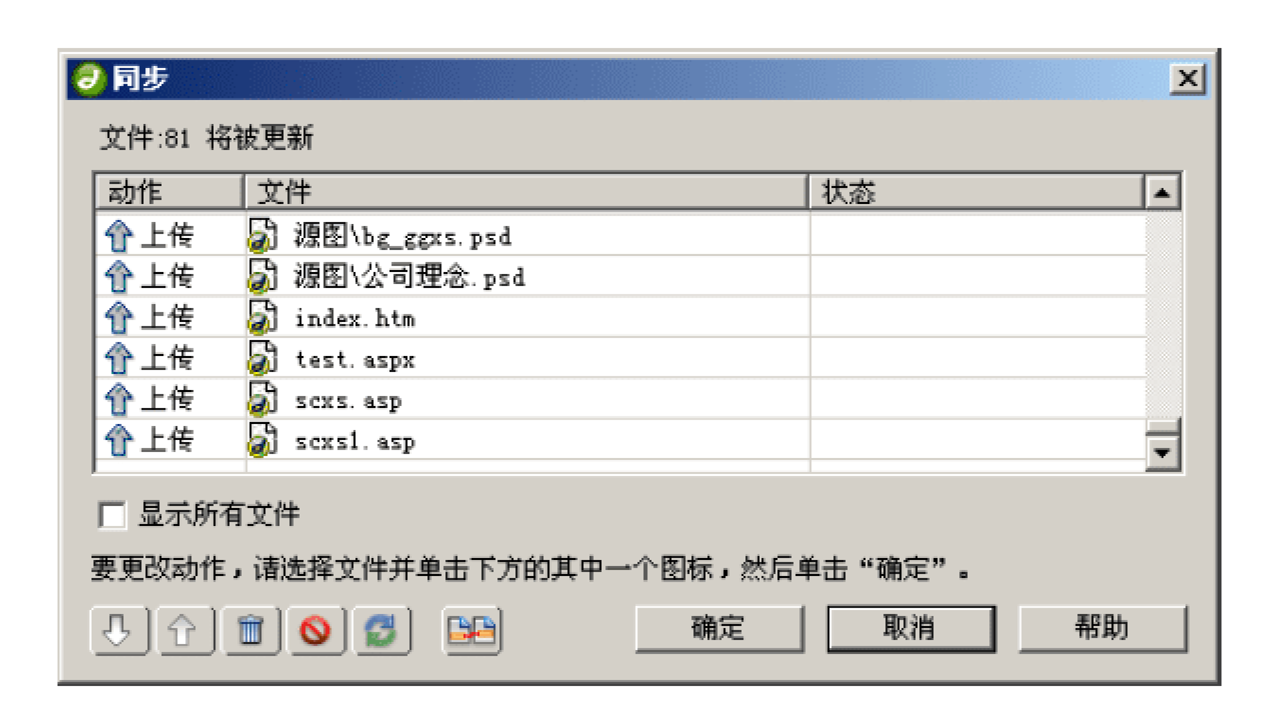


图 2.35　【同步】对话框

(3) 如果想在此次同步操作中忽略某个文件，可选择该文件，单击按钮。在确认所有列出来的要上传或获取的文件之后，单击【确定】按钮，即可执行同步文件操作。

2.4.3　检查站点链接

检查站点链接功能可帮助我们在整个站点中检查断掉的链接、外部链接以及孤立文件。在这里，断掉的链接是指链接文件在本地磁盘中没有找到，外部链接是指链接到站点外面的链接，孤立文件是指未被链接或引用的文件。

如要检查站点链接，可选择【站点】|【检查站点范围的链接】命令，系统将对所有链接进行检查，并将检查结果显示在【结果】面板组中的【链接检查器】面板中，如图 2.36 所示。

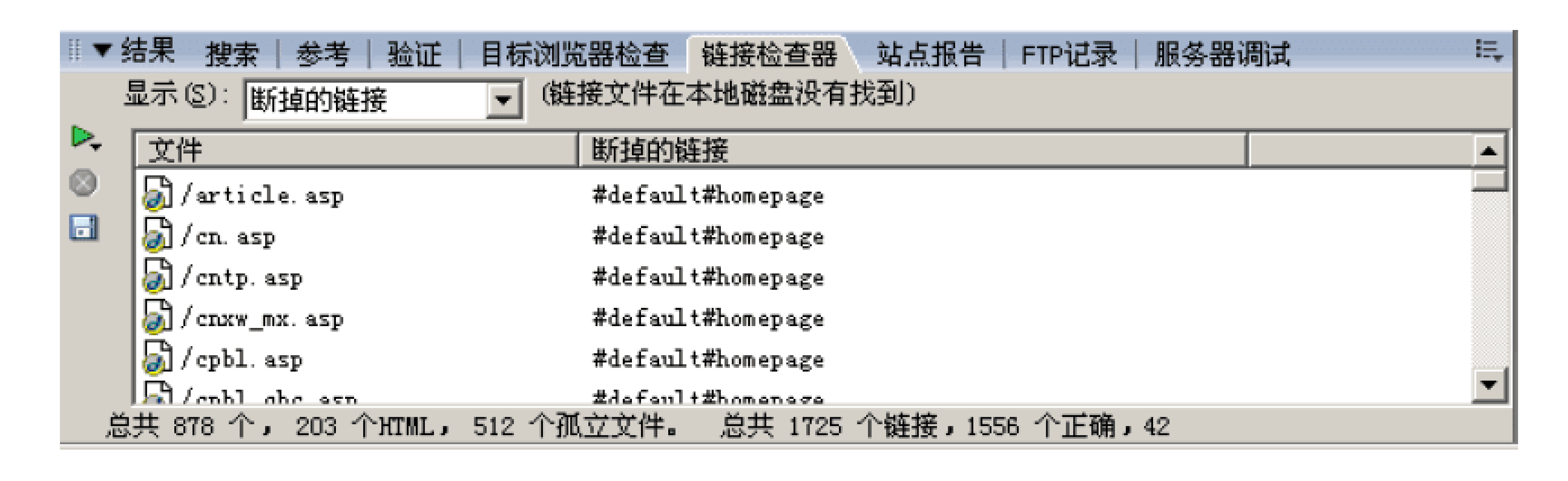


图 2.36　链接检查结果

默认情况下，在【链接检查器】面板中显示的是断掉的链接。从【显示】下拉列表框中选择【外部链接】选项或【孤立文件】选项，可分别查看相应类别的检查结果，如图 2.37 和图 2.38 所示。

图 2.37　检查结果——外部链接

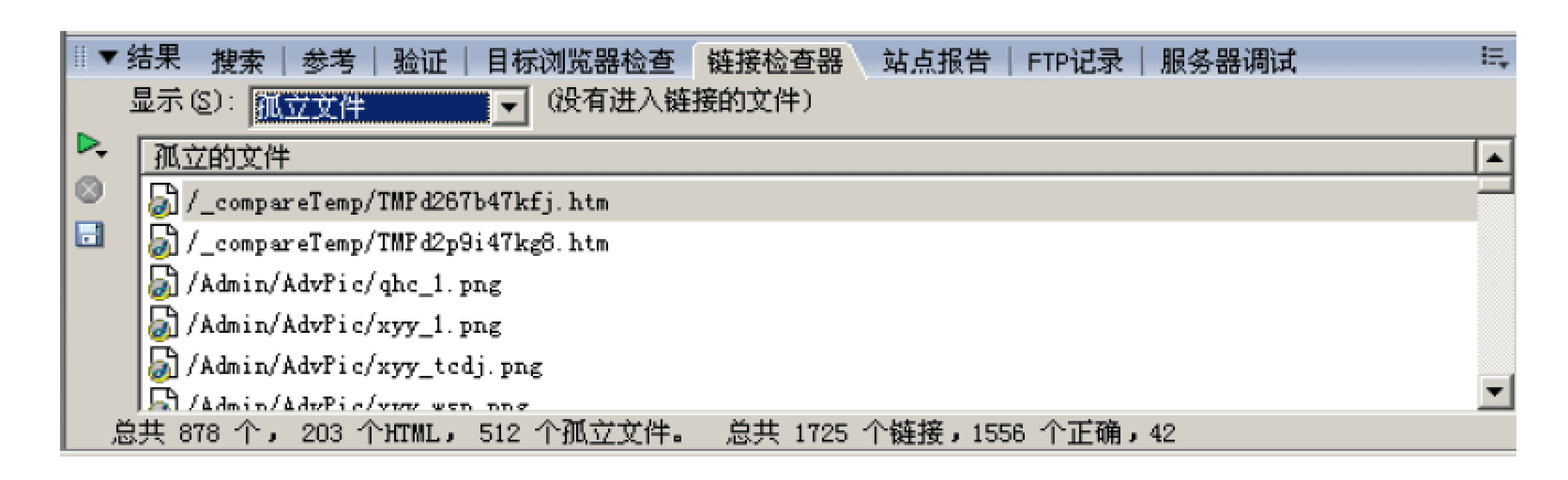


图 2.38　检查结果——孤立文件

单击按钮，可重新检查链接，其可选操作包括 3 种：检查当前文档中的链接、检查当前整个本地站点的链接和检查站点中所选文件的链接。单击按钮，可将当前检查结果保存为文本文件。

除了检查站点链接外，还可检查当前文档的链接和检查指定文件/文件夹的链接。对于前者，可选择【文件】|【检查页】|【检查链接】命令；对于后者，可在【文件】面板中，选择要检查链接的文件或文件夹，然后右击，在弹出的快捷菜单中选择【检查链接】|【选择文件/文件夹】命令。

通过检查链接，可检查链接的有效性和完整性。

2.4.4　站点报告

在 Dreamweaver 中，可对当前文档、选定文件或整个站点的工作流程及 HTML 属性(包括辅助功能)运行站点报告。

选择【站点】|【报告】命令，此时将弹出【报告】对话框，如图 2.39 所示。

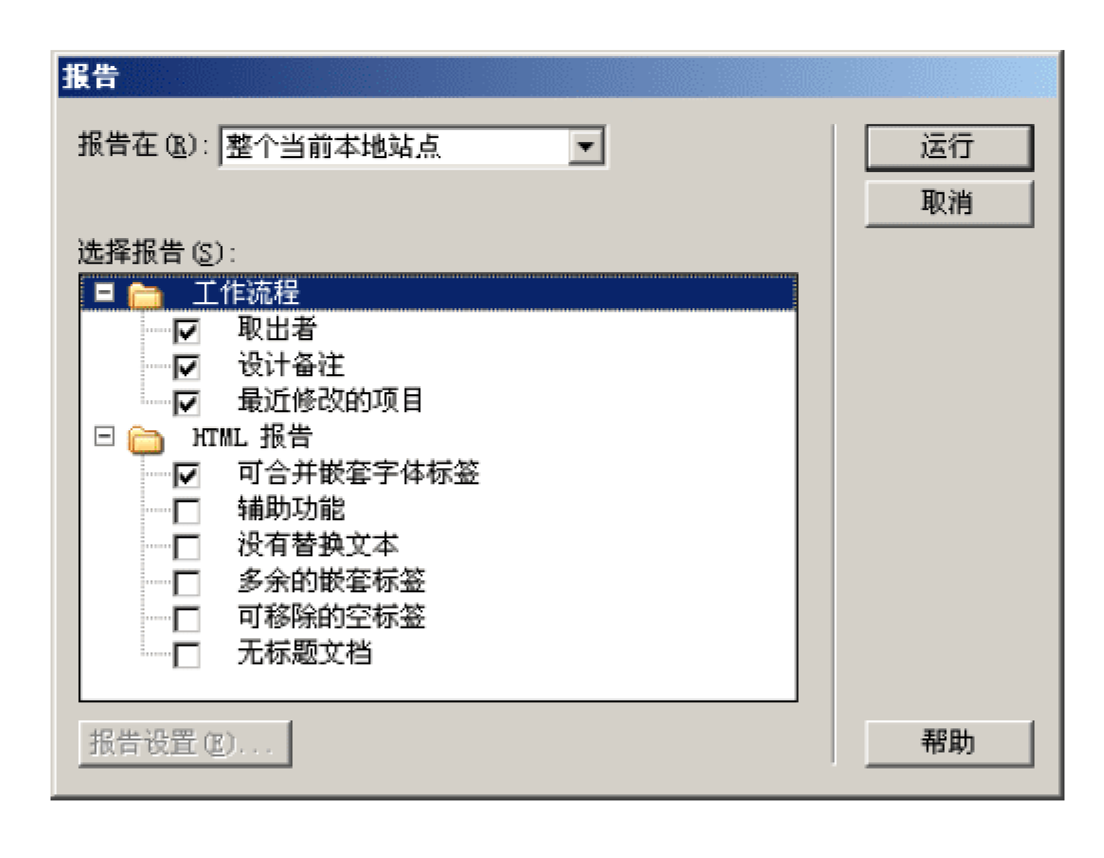


图 2.39　【报告】对话框

在【报告在】下拉列表框中，可选择运行报告的范围，其选项包括【当前文档】、【整个当前本地站点】、【站点中的已选文件】以及【文件夹】等。

在报告类型中，分为了两大类：【工作流程】和【HTML 报告】。

工作流程报告可以帮助用户改进站点设计小组中各成员之间的协作，其内容包括【取出者】、【设计备注】、【最近修改的项目】。其中，【取出者】将会列出谁取出了哪些文件，【设计备注】将会列出哪些文件具有与之关联的设计备注信息，【最近修改的项目】将会列出最近修改了哪些文件。选择其中一项，单击【报告设置】按钮，可对其进行更为详细的过滤设置。图 2.40 所示的图即为【最近修改的项目】一项的过滤设置对话框。

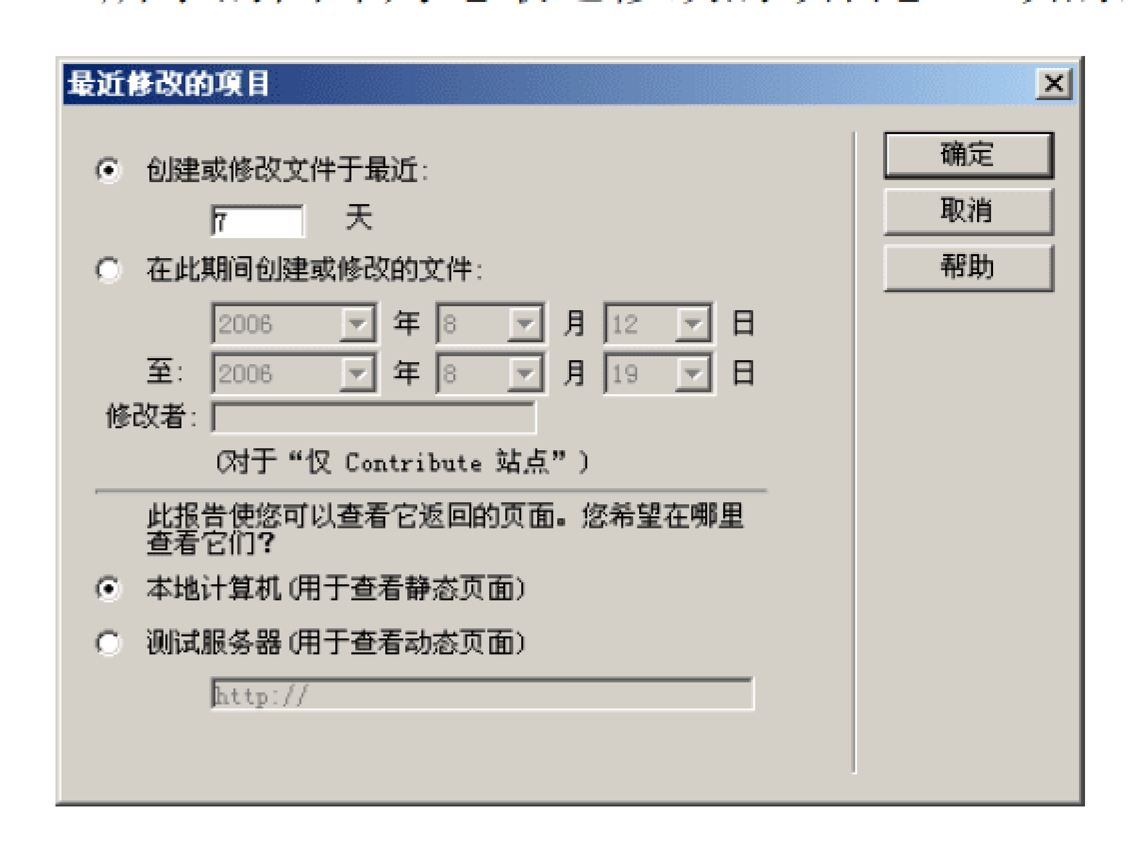


图 2.40　【最近修改的项目】对话框

提示：只有在定义远程端点后，才能运行工作流程报告。

HTML 报告可帮助我们对多个 HTML 属性编辑和生成报告，它还可以帮助用户检查可合并嵌套字体标签、辅助功能、没有替换文本、多余的嵌套标签、可移除的空标签以及无标题文档等。

- 【可合并嵌套字体标签】：列出所有可以为清理代码而合并的嵌套字体标签。
- 【辅助功能】：列出文档的内容与内置的辅助功能准则之间的冲突。
- 【没有替换文本】：列出所有没有设置替换文本的 img 标签。
- 【多余的嵌套标签】：列出冗余的可清理的嵌套标签。
- 【可移除的空标签】：列出可移除的空标签。
- 【无标题文档】：列出所有无标题的文档。

在 HTML 报告中，可对【辅助功能】一项进行详细的报告设置，如图 2.41 所示。

在【报告】对话框中，选择所要运行的报告类别，单击【运行】按钮，系统即可对报告范围内的所有文件进行检测，并生成报告信息，显示在【结果】面板组中的【站点报告】面板中。图 2.42 和图 2.43 显示的分别为工作流程报告和 HTML 报告的结果。

在运行完报告后，单击按钮，可将报告保存为 XML 文件供日后参考。

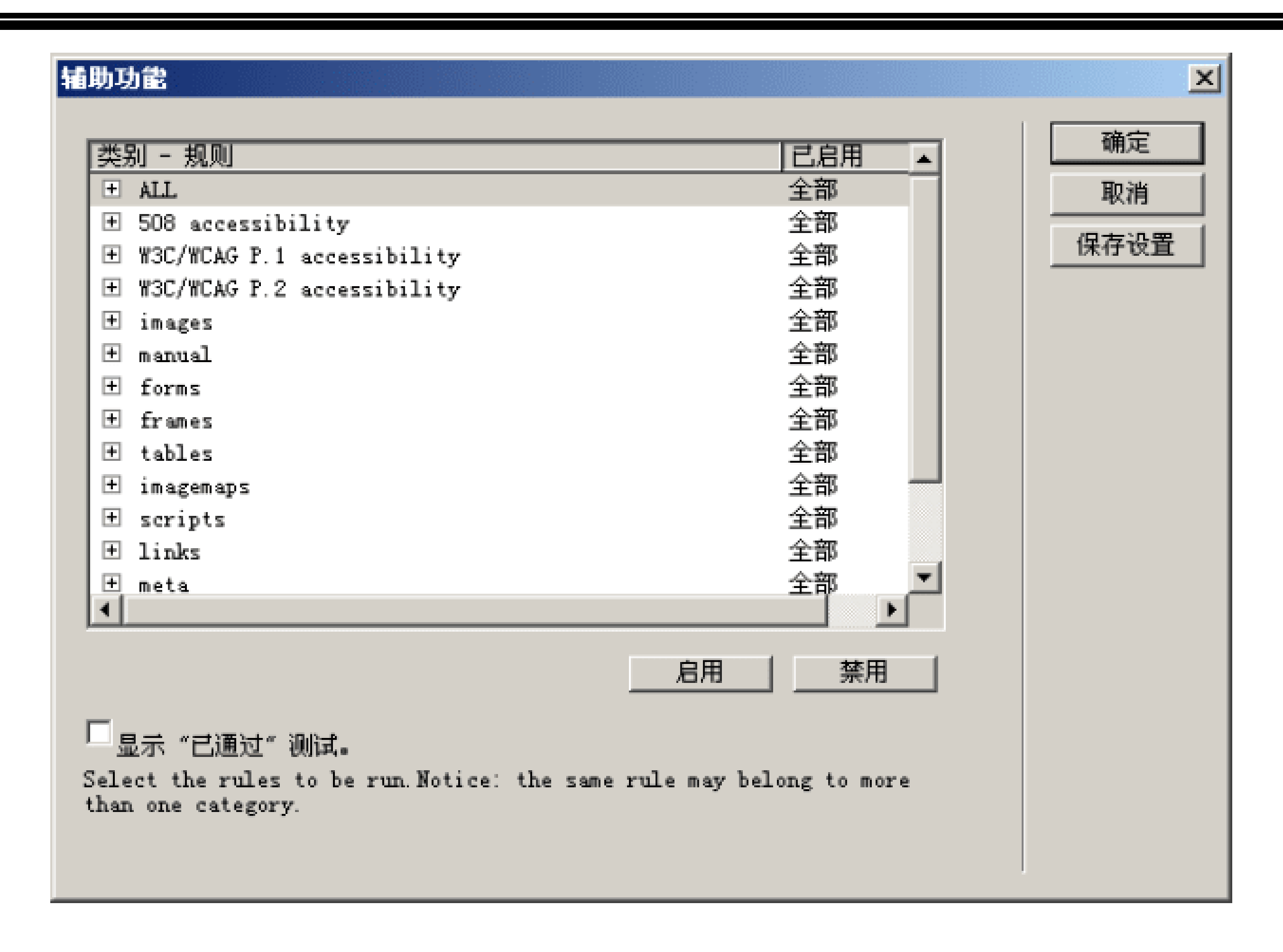


图 2.41　【辅助功能】对话框

图 2.42　工作流程报告

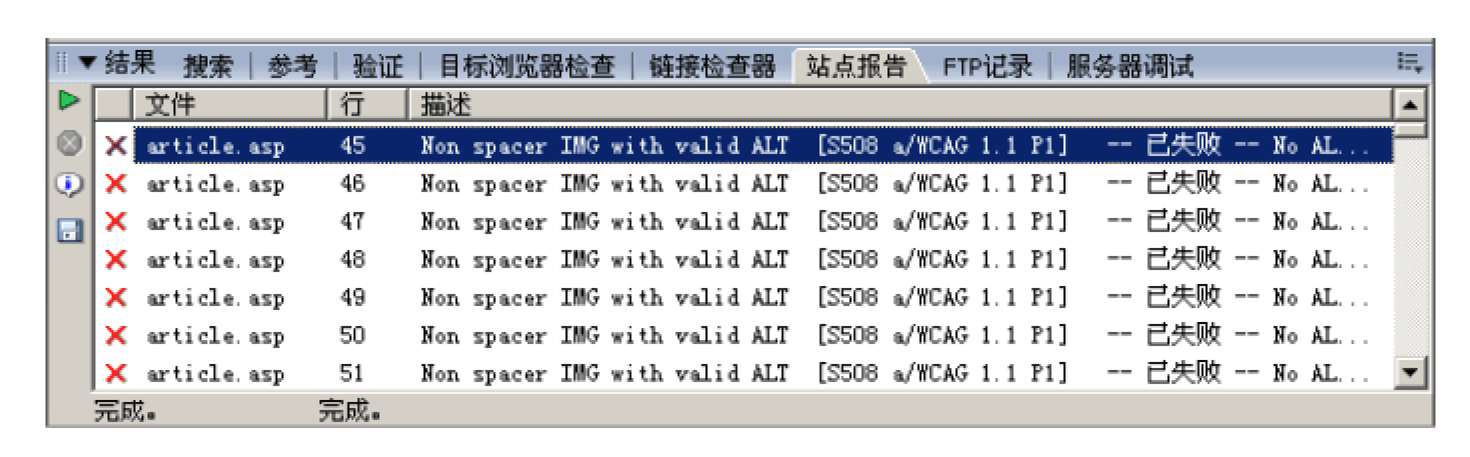


图 2.43　HTML 报告

2.5　习　　题

(1) 了解新建站点的整个过程，重点掌握站点设置中本地信息、远程信息和测试服务器等三项信息的具体设置，以及其中各个选项的含义。

(2) 掌握站点管理中，对站点文件的有效链接和失效链接的检测。

(3) 掌握本地站点与远程站点之间的文件传递操作，包括获取、取出、上传、存回等相关操作。

第 3 章　Dreamweaver 8 的基本操作

通过前面章节的学习，读者对 Dreamweaver 8 已经有了一个初步的认识。在本章中，将介绍 Dreamweaver 8 中一些基本元素的相关操作，包括文本、图像、链接的处理以及表格的具体使用。在最后的典型实例中，将通过一个博客首页面的制作来帮助大家了解各种元素的综合应用。

3.1　添 加 文 本

文本是网页的一个最基本的组成部分，也是网页中表达信息的一个主要途径。对文本的处理是网页设计中的一个最简单、最基础的操作。

在 ASP.NET 页面中，文本可分为静态文本和动态文本。静态文本是指直接在页面中输入的文本，其浏览时与输入时一致；动态文本是指通过表达式或函数所返回的文本，或是与指定的数据源中的字段绑定在一起的文本。动态文本的输出显示是不确定的，而是动态改变的。本节主要介绍静态文本的相关操作，动态文本的处理与静态文本的处理是一样的。

3.1.1　设置文本格式

在 Dreamweaver 8 中，对文本格式的设置主要是通过【属性】面板来实现的。在页面中，选择需要设置格式的文本，在【属性】面板中即可显示相应的格式项目，如图 3.1 所示。

图 3.1　【属性】面板

1. 字体

当在 Dreamweaver 8 的编辑区中输入文本时，系统将默认的字体(宋体)应用到文本中。如果需要更改文本使用的字体，则可先选择文本，然后在【属性】面板中单击【字体】下拉列表框，此时将弹出【字体】下拉列表，如图 3.2 所示。

在该下拉列表中，显示了 Dreamweaver 8 中默认提供的常用字体列表，而其中大部分是英文字体。选择【字体】下拉列表中的【编辑字体列表】选项，将弹出【编辑字体列表】对话框，如图 3.3 所示。

图 3.2　【字体】下拉列表

图 3.3　【编辑字体列表】对话框

在【可用字体】列表框中，显示了当前操作系统中安装的所有字体。如果需要将其中的字体添加至【字体列表】列表框中，可选择该字体，然后单击«按钮，将其添加至【选择的字体】列表框中，同时将其添加至【字体列表】中；如果需要将指定的字体从【选择的字体】列表框中删除，可先在【选择的字体】列表框中选择该字体，然后单击»按钮即可。

单击【字体列表】列表框左侧的+按钮，可以再次在【字体列表】列表框中添加新的字体；而单击−按钮，则可以在字体列表中删除所选择的字体。单击【字体列表】列表框右侧的▲按钮，可以将所选择的字体在字体列表中上移一个位置；而单击▼按钮，则可以将所选择的字体在字体列表中下移一个位置。

单击【确定】按钮，即可完成字体列表的编辑设置。

提示： 在页面设计中，应尽可能少地使用不常见的字体。因为，如果客户端没有安装当前页面所设置的字体，则将以默认的字体进行显示，从而失去了字体设置的意义。

2. 大小

在页面设计中，常常需要根据情况设置文本的列表大小。在 Dreamweaver 8 中，对文本大小的设置是通过【属性】面板中的【大小】下拉列表框来实现的，如图 3.4 所示。

在【大小】下拉列表框中，提供了两种类型的文字大小设置：绝对大小和相对大小。绝对大小是指固定的文字大小，图 3.4 所表示的数字即为文字的绝对大小，其中数字越小相应的文字就越小；而相对大小则是指相对于页面默认文字大小的比较。在默认情况下，页面的基准字号为 3，可以通过页面的<BaseFont>标签来改变默认的文字大小。一般来说，文字的绝对大小使用更为频繁。当选择了文字的绝对大小后，在其后面的下拉列表框中，将会显示所使用的字号单位，如图 3.5 所示。默认情况下，所使用的字号单位为“像素”(px)，用户也可根据需要选择其他的字号单位。

图 3.4　【大小】下拉列表

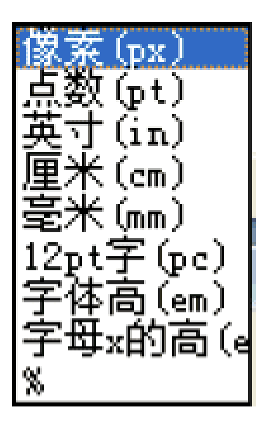


图 3.5　字号单位

提示： 在【大小】下拉列表框中，用户可直接输入数字来设置文字的绝对大小。比如，在【大小】下拉列表框中不能选择 13 作为字号，但可以直接在下拉列表框中输入数字 13。

3. 颜色

一般来说，页面背景设为白色和文字设为黑色。如果需要获取其他的效果，则可对文字的颜色进行设置。

选择要设置颜色的文本，单击【属性】面板中的【文本颜色】按钮，即可弹出【立方色】调色板，如图 3.6 所示。

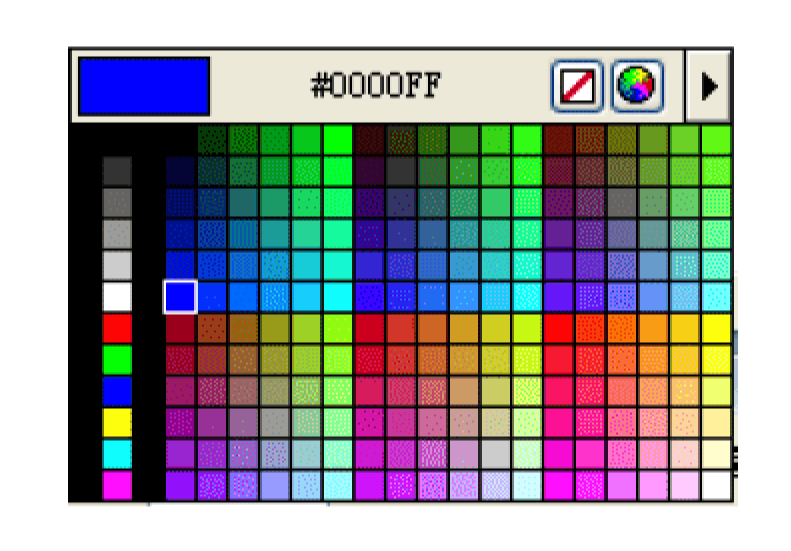


图 3.6　【立方色】调色板

当将鼠标移动到相应的色块上时，鼠标指针将变成吸管形状，同时在调色板的上方显示当前颜色所对应的十六进制值。单击，即可选择该颜色。

单击调色板中的【默认颜色】按钮，可清除所选择的颜色，恢复成默认的颜色。

单击调色板中的【系统颜色拾取器】按钮，则可弹出如图 3.7 所示的【颜色】对话框。

图 3.7　【颜色】对话框

在此对话框中，除了可以选择系统的基本颜色外，还可以自定义颜色。用户可以在右侧的色域面板中选择自定义的颜色，并通过亮度滑块调节自定义颜色的亮度值。单击【添加到自定义颜色】按钮，可将当前所设置的自定义颜色添加到左侧的【自定义颜色】区域的颜色框中。单击【确定】按钮，就完成了自定义颜色的添加。

除了以上可选择的颜色外，Dreamweaver 还通过其他的颜色类型为用户提供了更多的颜色选择。单击调色板中的▶按钮，可打开调色板选择菜单，如图 3.8 所示。

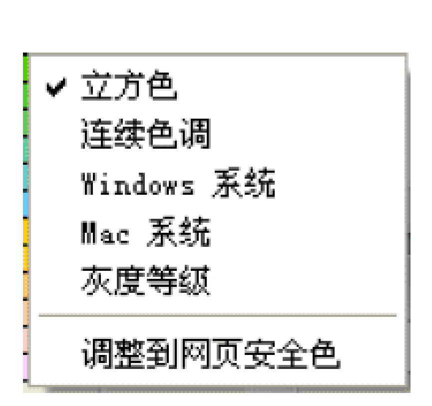


图 3.8　调色板选择菜单

其中，提供了更多类型的调色板，除了默认【立方色】调色板外，还包括【连续色调】、【Window 系统】、【Mac 系统】、【灰度等级】等调色板。在图 3.9～图 3.12 中，显示了以上类型调色板分别对应的可选颜色样式。

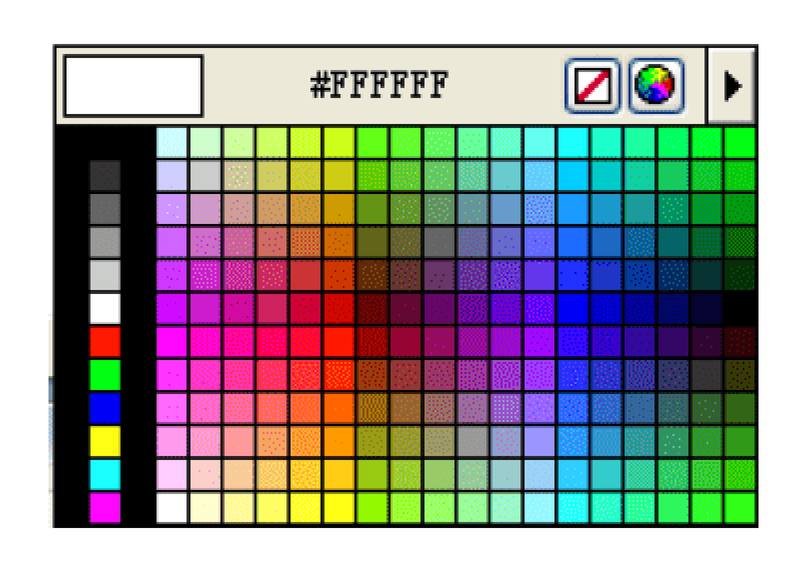


图 3.9　【连续色调】调色板

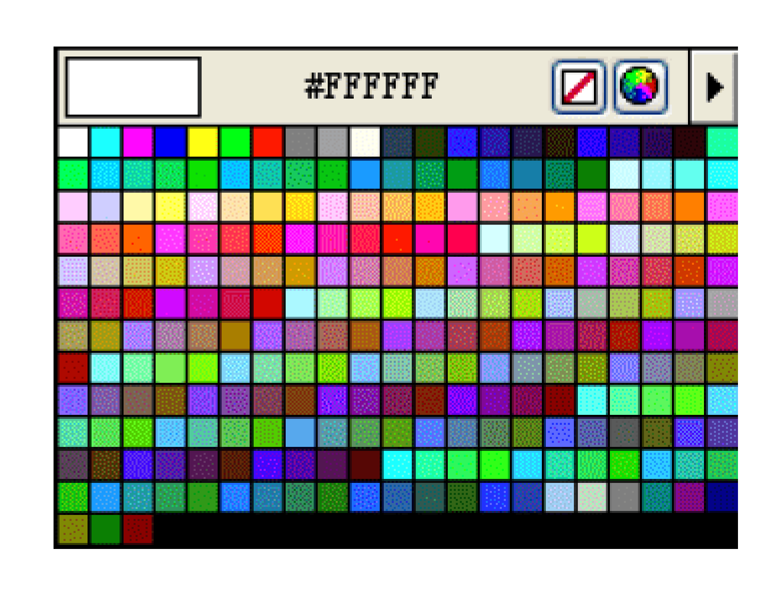


图 3.10　【Window 系统】调色板

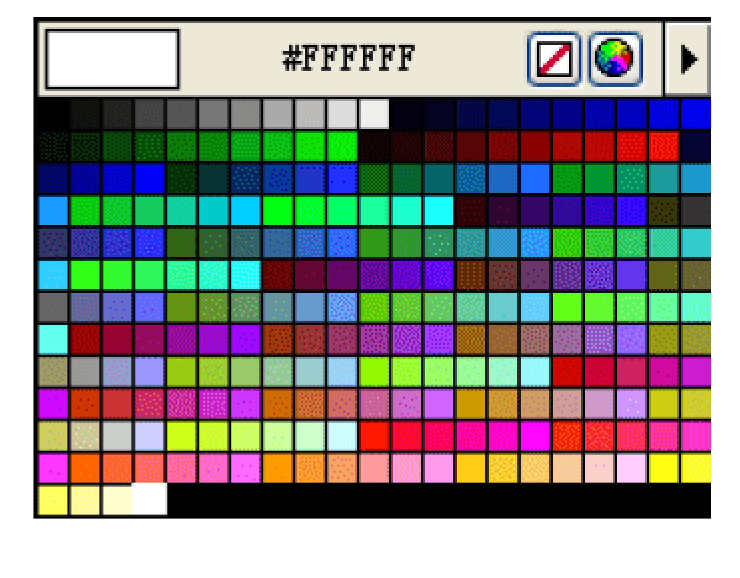


图 3.11　【Mac 系统】调色板

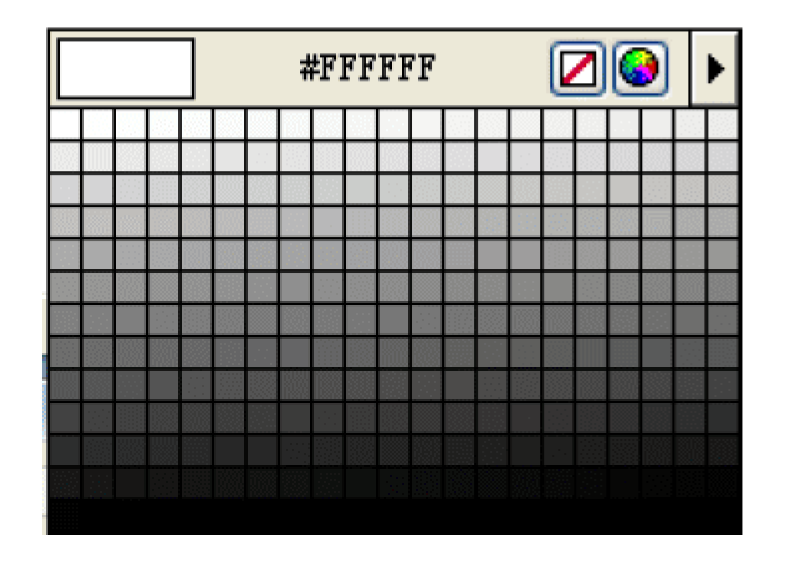


图 3.12　【灰度等级】调色板

除了通过调色板来选择颜色外，还有一种更直接的办法，那就是在▢按钮后的文本框中直接输入颜色所对应的十六进制数值。

4. 文本样式

在文本格式的设置中，除了以上提到的字体、大小、颜色外，还可设置文本的其他样式属性，如加粗、斜体、下划线、删除线等。除了最常用的加粗、斜体样式可直接在【属性】面板中通过单击相应的按钮(**B** 和 *I*)实现外，其余样式均可通过【文本】|【样式】子菜单中的命令来实现，如图 3.13 所示。

粗体(B)　Ctrl+B
斜体(I)　Ctrl+I
下划线(U)
删除线(S)
打字型(T)
强调(E)
加强(R)
代码(C)
变量(V)
范例(A)
键盘(K)
引用(Q)
定义(D)
已删除(L)
已插入(N)

图 3.13　【样式】子菜单中的命令

其中，各个命令的含义及所对应的<HTML>标记见

表 3.1。

表 3.1　【样式】子菜单中的命令及功能

样　式	功　能	对应的<HTML>标记
粗体	将选中的文本加粗显示	
斜体	将选中的文本变斜	<i></i>
下划线	为选中的文本添加下划线	<u></u>
删除线	为选中的文本添加删除线	<s></s>
打字型	为选中的文本添加等宽标记	<tt></tt>
强调	为选中的文本使用斜体着重强调	
加强	为选中的文本使用黑体着重强调	
代码	为选中的文本使用描述程序代码效果	<code></code>
变量	为选中的文本使用动态字体	<var></var>
范例	为选中的文本使用标准字体	<samp></samp>
键盘	为选中的文本使用黑体	<kbd></kbd>
引用	为选中的文本使用斜体	<cite></cite>
定义	为选中的文本使用斜体	<dfn></dfn>
已删除	为选中的文本添加删除线	<del></del>
已插入	为选中的文本添加下划线	<ins></ins>

3.1.2　设置段落

段落是排列文本时最常用的格式之一，通常情况下，页面中的大量文本都是由段落组成的。在 HTML 中，开始标记<P>和结束标记</P>之间的文本被标识为一个段落。对于段落的创建，只需在分段的地方按 Enter 键即可。

对于段落的格式，主要包括对齐设置、缩进/凸出设置以及标题设置等几个方面。

1. 段落对齐

在【属性】面板中，提供了 4 种段落文本的对齐方式：(左对齐)、(居中对齐)、(右对齐)和(两端对齐)。其中，左对齐是指将段落内的所有文本行靠左对齐，居中对齐是指将段落内的所有文本行居中对齐，右对齐是指将段落内的所有文本靠右对齐，两端对齐则是指将段落内的所有文本行尽量向两端对齐。各种对齐方式的效果如图 3.14 所示。

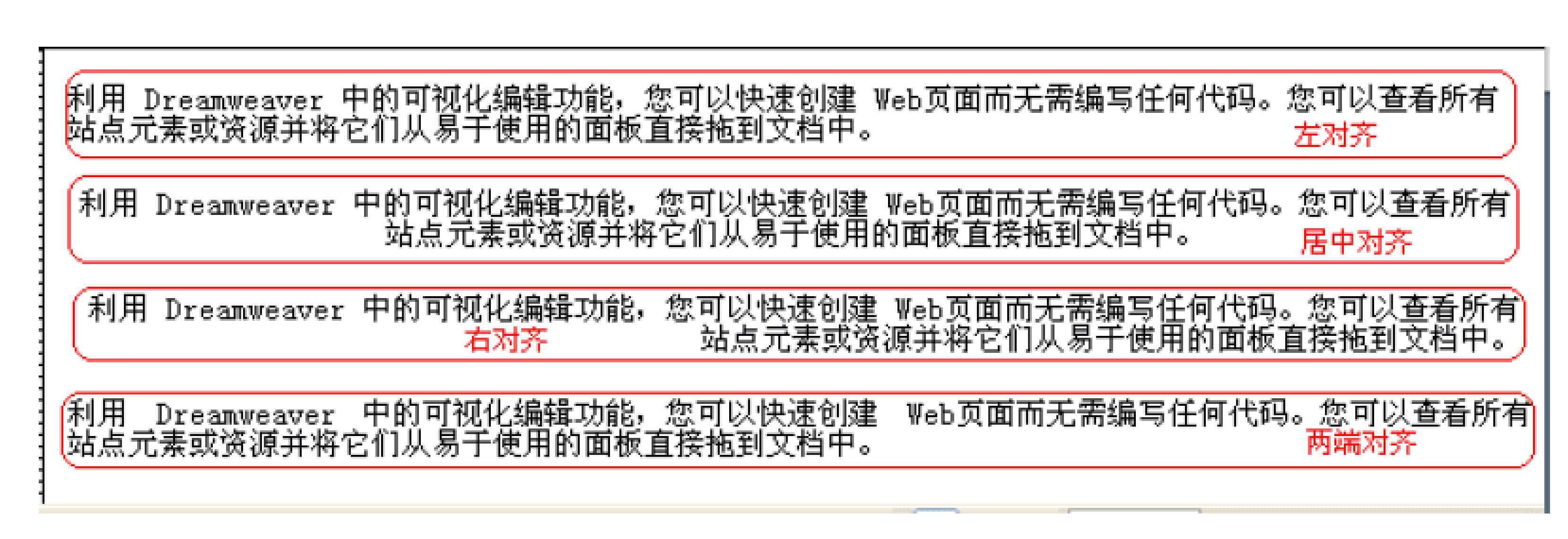


图 3.14　文本对齐效果

2. 段落缩进/凸出

段落的缩进是指缩进页面两侧的文本，而凸出则与缩进相反，用于取消段落的缩进操作。对于一个段落，可执行多重缩进。在<HTML>中，缩进操作所对应的标签为<blockquote>。在【属性】面板中，单击按钮，可执行段落的缩进操作；单击按钮，可执行段落的凸出操作。在图 3.15 中，显示了段落缩进与未缩进的效果对比。

利用 Dreamweaver 中的可视化编辑功能，您可以快速创建 Web页面而无需编写任何代码。您可以查看所有站点元素或资源并将它们从易于使用的面板直接拖到文档中。 缩进的段落

利用 Dreamweaver 中的可视化编辑功能，您可以快速创建 Web页面而无需编写任何代码。您可以查看所有站点元素或资源并将它们从易于使用的面板直接拖到文档中。 未缩进的段落

图 3.15　段落的缩进

3. 设置标题

为了便于将页面中的文本进行分级分类，在 Dreamweaver 中提供了为文本设置标题的功能，这样便于用户了解文本的层次结构。

在 Dreamweaver 8 中，选择要设置为标题的文本，然后单击【属性】面板中【格式】下拉列表框，此时将会显示如图 3.16 所示的下拉列表。

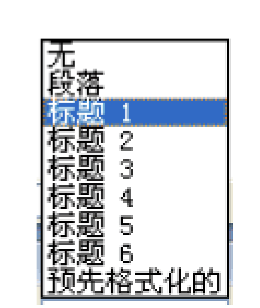


图 3.16　【格式】下拉列表

这里，提供了 6 种预定义的标题选项，其名称分别为“标题 1”～“标题 6”(由高到低的级别顺序)。对应的显示效果，如图 3.17 所示。

Dreamweaver 8 与 ASP.NET

Dreamweaver 8 与 ASP.NET

Dreamweaver 8 与 ASP.NET

Dreamweaver 8 与 ASP.NET

Dreamweaver 8 与 ASP.NET

Dreamweaver 8 与 ASP.NET

图 3.17　标题列表

在图中，第 1 行～第 6 行，依次对应于“标题 1”～“标题 6”。虽然，标题的格式是默认的，但并不是固定的。在将文本设置为标题之后，用户可以再对该标题的文本格式进行设置，包括字体类型、大小、颜色以及加粗、斜体等相关样式的设置。

3.1.3　设置列表

在 Dreamweaver 8 中，可以创建 3 种形式的列表：项目列表、编号列表和定义列表。

项目列表又叫无序列表，它将所要列举的项目一一列出，而不进行任何编号；编号列

表，又叫有序列表，它是将所要列举的项目按顺序进行排列，并依次进行编号；定义列表则用于对文本中的某一个词或某一段文字进行解释，显示该词或文字的说明信息。在 Dreamweaver 8 的【属性】面板中，提供了对项目列表和编号列表的快捷操作按钮。

选择需要创建列表项的文本，单击【属性】面板中的按钮，即可创建一个项目列表，如图 3.18 所示。

在项目列表的文本之后，按 Enter 键，可生成一行新的项目列表项；如果在项目列表的文本之后按两次 Enter 键，则可结束当前的项目列表。在默认情况下，项目列表的项目符号将采用黑色实心圆点，也可更改默认的项目符号。

将光标置于项目列表的文本中，单击【属性】面板中的【列表项目】按钮，此时将弹出【列表属性】对话框，如图 3.19 所示。

图 3.18　项目列表

图 3.19　【列表属性】对话框

在【列表类型】下拉列表框中，提供了两种符号类型选项：项目符号和正方形。项目符号即系统默认的黑色实心圆点，正方形则表示黑色实心方块，用户可根据需要来选择。

在页面上选择需要创建列表项的文本，单击【属性】面板中的按钮，即可创建一个编号列表，如图 3.20 所示。

默认情况下，编号列表的项目符号将采用数字。与项目列表类似，同样可更改默认的项目符号。将光标置于编号列表的文本中，单击【属性】面板中的【列表项目】按钮，此时将弹出【列表属性】对话框，如图 3.21 所示。

图 3.20　编号列表

图 3.21　【列表属性】对话框

在【样式】下拉列表框中，提供了 5 种列表的符号样式，它们分别是数字、小写罗马字符、大写罗马字符、小写字母和大写字母，用户可根据需要自行选择。此外，在【开始计数】文本框中，可设置编号列表的起始编号。

在项目列表和编号列表中，所对应的 HTML 标签分别为<ul>和<ol>，而其中的各具体项目所对应的 HTML 标签均为<li>。

定义列表主要用于对文本中的特定术语进行解释或说明，它由两部分组成：定义名称和定义内容。其中，定义内容位于定义名称的下一行，且缩进显示。

创建的定义列表操作步骤如下。

(1) 在页面上输入定义名称。

(2) 选择【文本】|【列表】|【定义列表】命令。

(3) 将光标置于定义名称之后，按 Enter 键，此时将会缩进定义名称的下一行。

(4) 在定义名称的下一行中，输入定义内容即可。

定义列表的创建结果如图 3.22 所示。

博客

又叫Blog，它的英文全称为WebBlog，原意为“网络日记”。但是，随着网络的发展，博客已不仅仅是个人的网络日记，它已经和电子邮件、QQ、MSN一样，发展为互联网上最时尚的个人展示、网友交流工具，并成为当今最精彩的网络生活方式之一

图 3.22　定义列表

整个定义列表所对应的<HTML>标签为<dl>，其中，定义名称和定义内容所对应的<HTML>标签分别为<dt>和<dd>。

3.2　添 加 图 像

图像是网页中的一个非常重要的元素，它极大地丰富了页面的内容，同时带给人们不一样的视觉效果。

3.2.1　插入图像

在页面中插入图像有两种方法：一种方法是单击【常用-插入】工具栏中的【图像】按钮；另一种方法则是选择【插入】|【图像】命令。使用这两种方法中的任一种，系统都将弹出【选择图像源文件】对话框，如图 3.23 所示。

图 3.23　【选择图像源文件】对话框

在图 3.23 所示的对话框中，可通过浏览磁盘的文件夹来指定所要插入的图像文件。选

择相应的图像文件后，在对话框的右侧可以预览当前图像。单击【确定】按钮，即可将当前指定的图像文件插入至页面的光标所在处。此时，还将弹出【图像标签辅助功能属性】对话框，如图 3.24 所示。

图 3.24　【图像标签辅助功能属性】对话框

在此对话框中，可设置图像的替换文本和详细说明链接等信息，该信息将用于当图像无法正常显示时替换图像的提示。单击【确定】按钮，即可完成图像的插入操作，如图 3.25 所示。

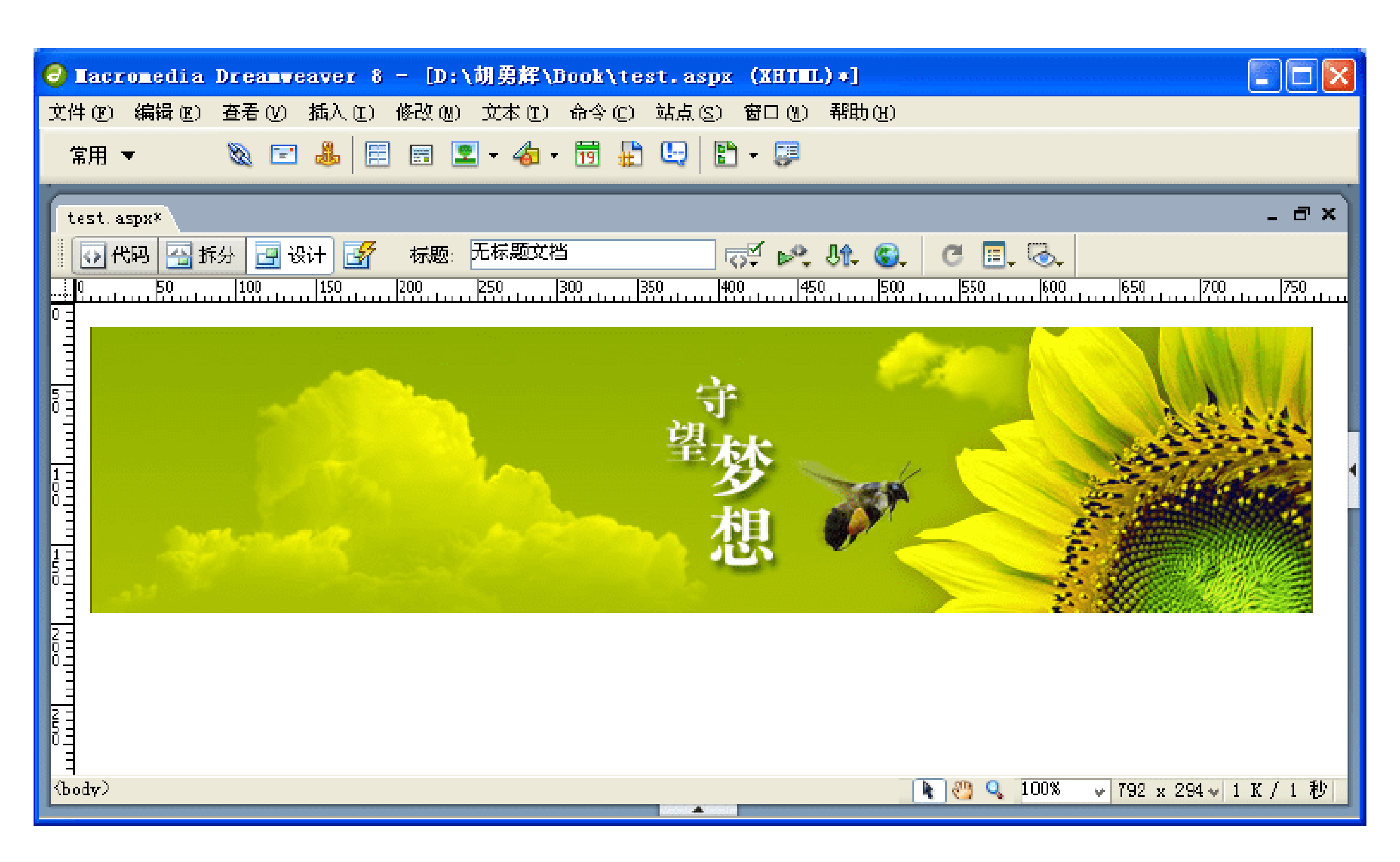


图 3.25　插入图像

3.2.2　设置图像属性

在插入图像后，可在【属性】面板中对图像的相关属性进行设置，如图 3.26 所示。

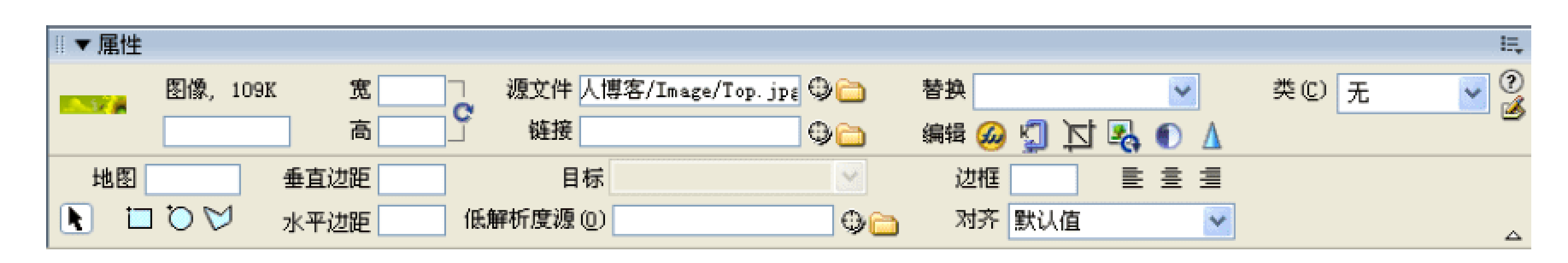


图 3.26　设置图像属性

1. 图像大小

在图像的【属性】面板中，【宽】和【高】文本框分别用于设置图像的显示大小。默认情况下，在插入图像时，图像将以 100%的比例(即原尺寸大小)显示。但在实际应用中，根据页面布局的需要，经常需要对图像的显示大小进行调整。“宽”决定了图像的显示宽

度，“高”则决定了图像的显示高度。直接在【宽】和【高】文本框中输入相应的数值，就能将图像的显示调整为自己所需要的尺寸，这里的数值单位默认为像素。单击按钮，可恢复图像的原始尺寸。

提示： 在此处对图像大小的设置，仅是改变其在屏幕上所显示的图像大小，而非改变图像的实际大小。如需改变图像的实际大小，则需使用专业的图像处理软件(如 Photoshop 等)来进行处理。

2. 图像边距

对于图像与周边文本或其他对象之间的间距，可通过【属性】面板中的【垂直边距】和【水平边距】选项来实现。【垂直边距】选项用于设置图像的顶部和底部边距，即设置图像与上、下文本之间的间距；【水平边距】用于设置图像的左侧和右侧边距，即设置图像与左、右文本之间的间距；两者的单位均默认为像素。

3. 图像源文件

在【源文件】文本框中，显示当前图像的路径及名称。通过单击【文件夹】按钮，将弹出的【选择图像源文件】对话框，在对该对话框中可重新设置图像的源文件。

4. 低解析度源

【属性】面板中的【低解析度源】文本框用于设置当载入主图像之前所需载入的图像。如果页面中所插入的图像较大，将导致页面显示很慢。此时，可通过【低解析度源】文本框来设置一个较小的替代图像，用来在加载主图像之前进行显示，以便提供一个缓冲的余地。一般来说，替代图像使用主图像的 2 位(黑和白)版本，因为它可以迅速载入并使访问者对他们等待看到的主图像有所了解。

设置替代图像的方法与设置图像源文件的方法完全相同，用户可通过在【低解析度源】文本框中直接输入替代图像的文件路径及文件名，也可通过单击【文件夹】按钮，在弹出的【选择文件】对话框中选择图像。

5. 图像边框

【边框】文本框用来为所显示的图像添加一个单色的矩形边框，边框的宽度以像素为单位，默认颜色为黑色。在默认情况下，图像无边框。

6. 图像对齐

【属性】面板中的【对齐】下拉列表框用于设置图像与同一行文本的对齐方式。在【对齐】下拉列表框中，提供了多种图像对齐方式，如图 3.27 所示。

- 【默认值】：浏览器默认的对齐方式，大多数的浏览器都采用基线对齐作为默认对齐方式。
- 【基线】：图像与当前行的文本的基线对齐。
- 【顶端】：图像的顶端与当前行中最高对象(图像或文本)的顶端对齐。

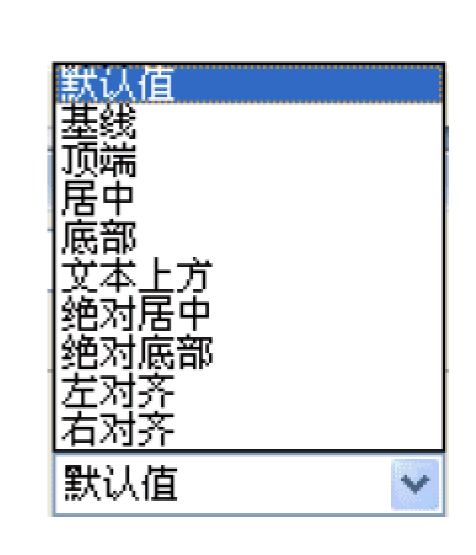


图 3.27　图像对齐方式

- 【居中】：图像的中部与当前行的基线对齐。
- 【底部】：图像底部与当前行中最低对象(图像或文本)的底部对齐。
- 【文本上方】：图像顶端与当前行中最高字符的顶端对齐。
- 【绝对居中】：图像的中部与当前行中文本的中部对齐。
- 【绝对底部】：图像的底部与文本行的底部对齐。
- 【左对齐】：当前图像放置在左边，文本在图像的右侧换行显示。
- 【右对齐】：当前图像放置在右边，文本在图像的左侧换行显示。

3.3　创 建 链 接

超级链接是网页中不可少的元素之一，它是网页之间相互关联的重要方式。根据链接的对象不同，超级链接的类型可分为文件链接、命名锚记链接和电子邮件链接 3 种。

3.3.1　文件链接

文件链接是指链接的目标对象为文件，而链接的源对象可以是文本、图像或其他对象。当选择需要创建链接的对象时，在【属性】面板中均包含有【链接】选项，如图 3.28 所示。

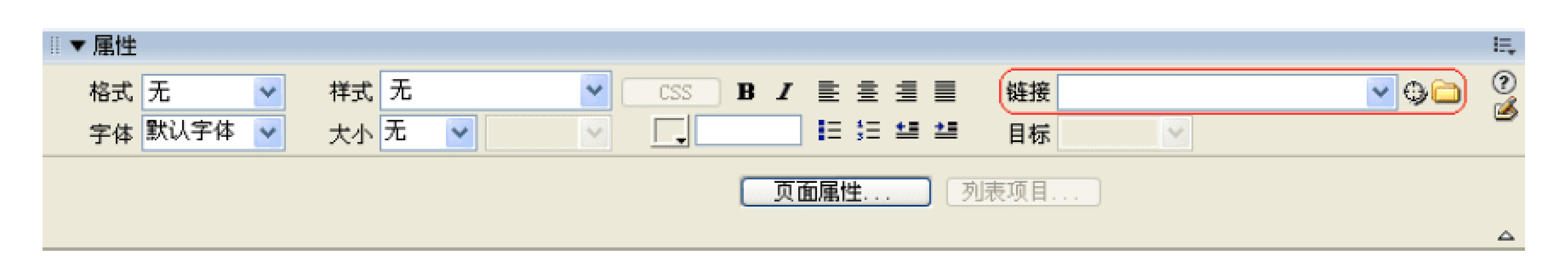


图 3.28　设置链接

在【属性】面板中，提供了 3 种方式来创建文件链接。

- 直接在【链接】文本框中输入链接的文件路径。
- 单击【链接】文本框右边的【文件夹】按钮，在弹出的【选择文件】对话框中，选择所要链接的文件，并单击【确定】按钮，如图 3.29 所示。

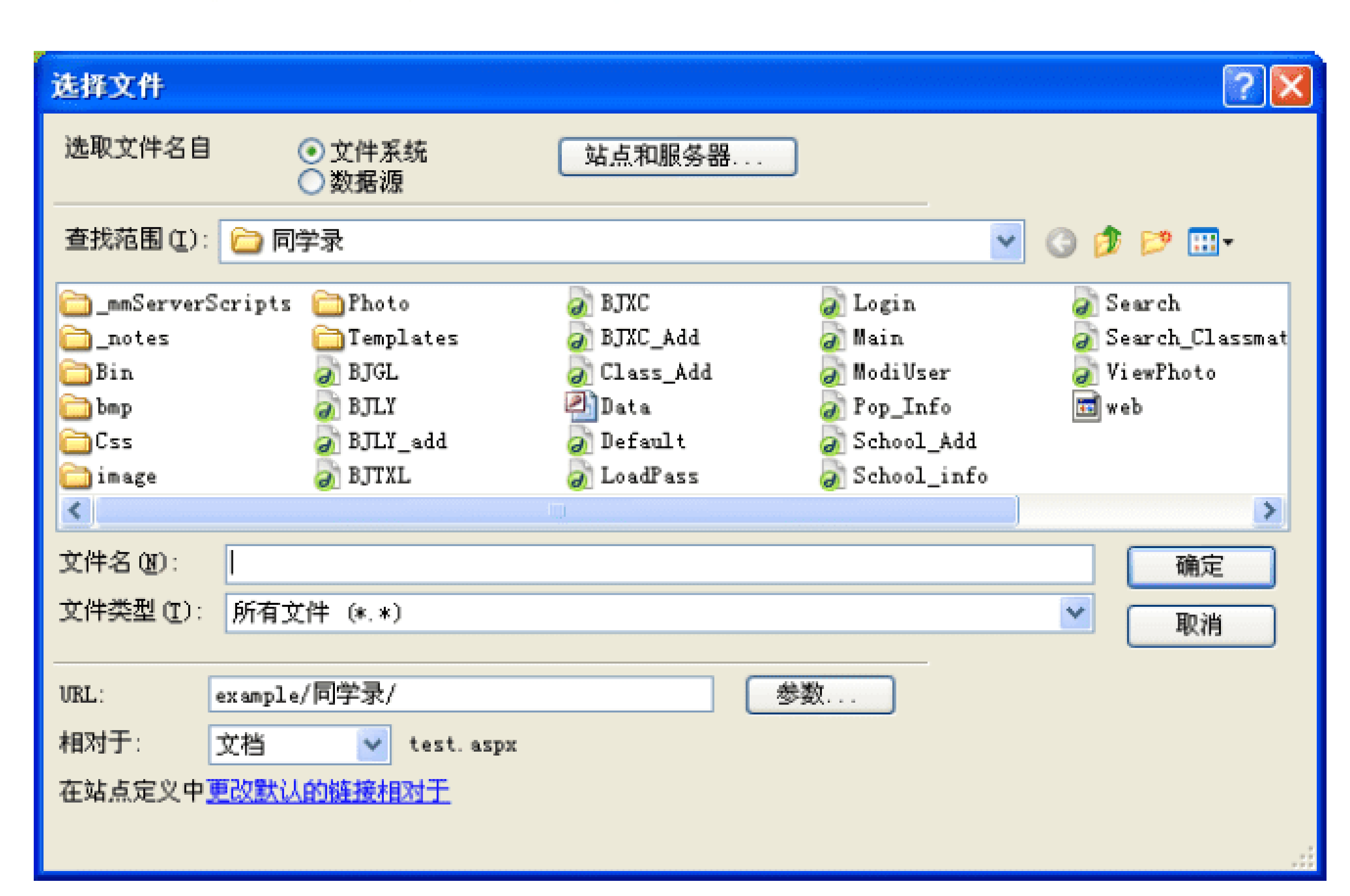


图 3.29　选择文件

- 如果所链接的文件在 Dreamweaver 中的【文件】面板中，则可按住【指向文件】

按钮，拖动鼠标至【文件】面板中所要链接的文件即可。

在【属性】面板中，还包括一项与链接相关的选项，那就是【目标】选项。当设置了文本或图像的超级链接时，该选项将显示为可操作项。在其下拉列表框中，包括_blank、_parent、_self、_top 等几个选项。其中，_blank 选项表示在新窗口打开链接页面，_parent 选项表示在当前页面的父窗口中打开链接页面，_self 选项表示在当前页面窗口中打开链接页面，_top 选项表示在当前页面最上一级的窗口中打开链接页面。

3.3.2　命名锚记链接

对于文件链接来说，只能实现页面与页面之间的链接。而如果需要在同一个页面内，设置不同的链接至页面中不同的地方，则需要使用命名锚记链接。

在创建命名锚记链接之前，首先需要定义命名锚记。选择需要创建锚记链接的文本，选择【插入】|【命名锚记】命令，此时将弹出【命名锚记】对话框，如图 3.30 所示。

图 3.30　【命名锚记】对话框

在【命名锚记】对话框中，输入锚记的名称。单击【确定】按钮，完成锚记的创建。此时，在定义锚记的地方将会显示相应的锚记符号。单击该锚记符号，【属性】面板将显示为锚，如图 3.31 所示。

图 3.31　设置锚记属性

在此面板中，可以更改所定义的锚记的名称。

命名锚记定义之后，则可将所要链接的对象指向定义好的锚记。

在页面中，选择命名锚记所要链接的对象，在【属性】面板的【链接】文本框中输入#Link1。其中，Link1 是前面所定义的命名锚记名称。

此时，命名锚记链接的创建已完成。在页面预览时，当单击设置命名锚记的文本后，页面将跳转至所定义的锚记处。

3.3.3　电子邮件链接

除了页面链接之外，在 Dreamweaver 中还可以创建电子邮件链接。当在页面中添加邮件链接之后，单击该链接即可调用系统默认的电子邮件客户端软件(如 Microsoft OutLook)，允许用户直接发送电子邮件。

电子邮件链接的创建相当简单，在页面上选择所要创建邮件链接的文本，选择【插入】|

【电子邮件链接】命令，此时将弹出【电子邮件链接】对话框，如图 3.32 所示。

在【文本】文本框中，将会显示在页面中所选择的要创建邮件链接的文本；在 E-Mail 文本框中，则可输入要链接的邮件地址。单击【确定】按钮，即可完成电子邮件链接的创建。

图 3.32　【电子邮件链接】对话框

在页面预览中，单击所创建的电子邮件链接，即可打开系统默认的电子邮件客户端软件，如图 3.33 所示。

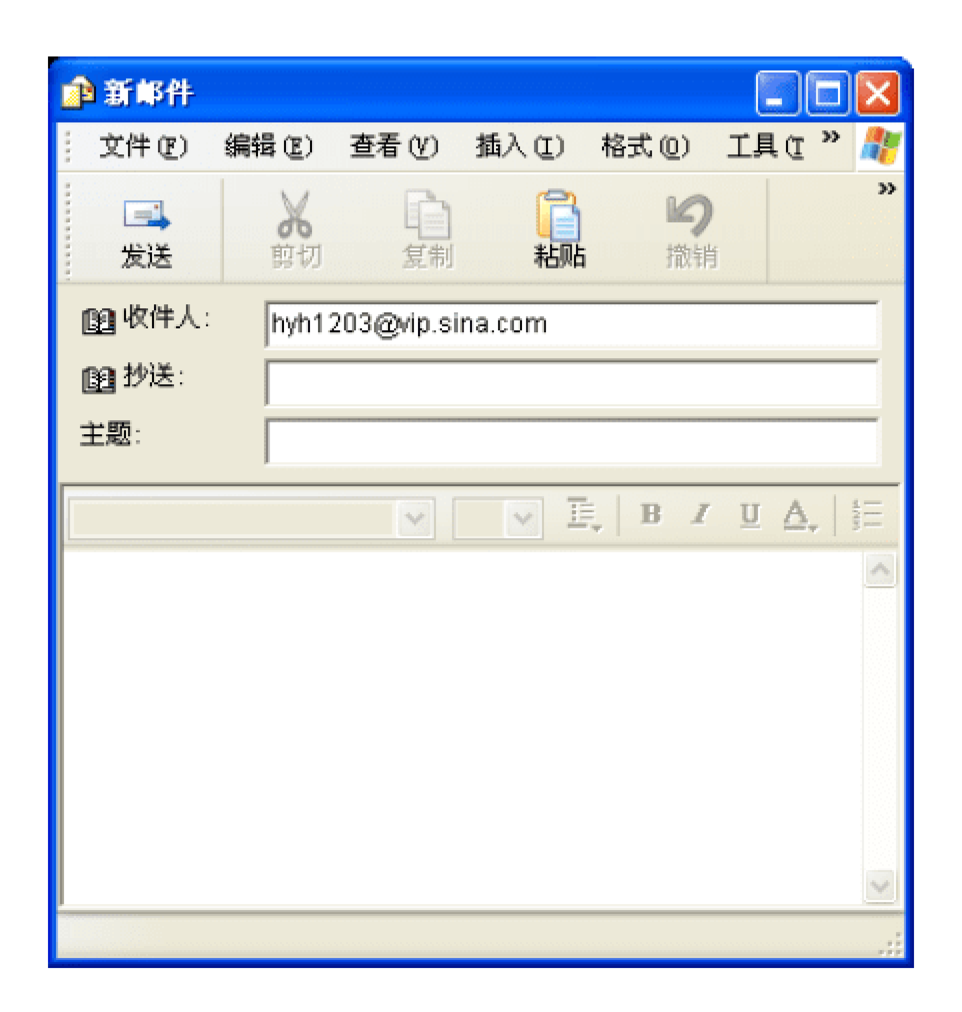


图 3.33　打开的系统默认的电子邮件客户端软件

其中，【收件人】文本框将自动显示为电子邮件链接所设置的邮件地址。用户只需输入邮件主题和邮件内容，单击【发送】按钮即可发送电子邮件。

3.4　使 用 表 格

表格是网页设计中用来控制页面布局的最常用工具之一，本节将介绍表格的具体使用。

3.4.1　创建表格

在页面中，将光标置于所需插入表格的地方，单击【常用-插入】工具栏中的【表格】按钮，将弹出【表格】对话框，如图 3.34 所示。

在此对话框中，提供了所要创建的表格的相关选项。其中，各选项的含义如下。

- 【行数】：用于设置表格所具有的行的数目。
- 【列数】：用于设置表格所具有的列的数目。

- 【表格宽度】：用于以像素为单位或按占浏览器窗口宽度的百分比指定表格的宽度。
- 【边框粗细】：用于设置表格边框的宽度，其单位为像素。
- 【单元格边距】：用于设置单元格边框与单元格内容之间的间距，其单位为像素。
- 【单元格间距】：用于设置单元格与邻近单元格之间的间距，其单位为像素。

图 3.34　【表格】对话框

- 【标题】：用于设置显示在表格外的表格标题。
- 【对齐标题】：用于设置表格标题相对于表格的对齐方式。
- 【摘要】：用于设置表格的摘要信息，该信息将不会显示在用户的浏览器中，但可通过屏幕阅读器读取。

设置相应的选项，单击【确定】按钮，即可完成表格的创建。

3.4.2　表格属性

对于已创建的表格，可以通过【属性】面板来设置表格的相关属性。在此之前，必须选择表格。将光标置于表格的任何一个单元格中，选择【修改】|【表格】|【选择表格】命令，此时将选定整个表格，同时【属性】面板中的选项变为设置表格属性的相关选项，如图 3.35 所示。

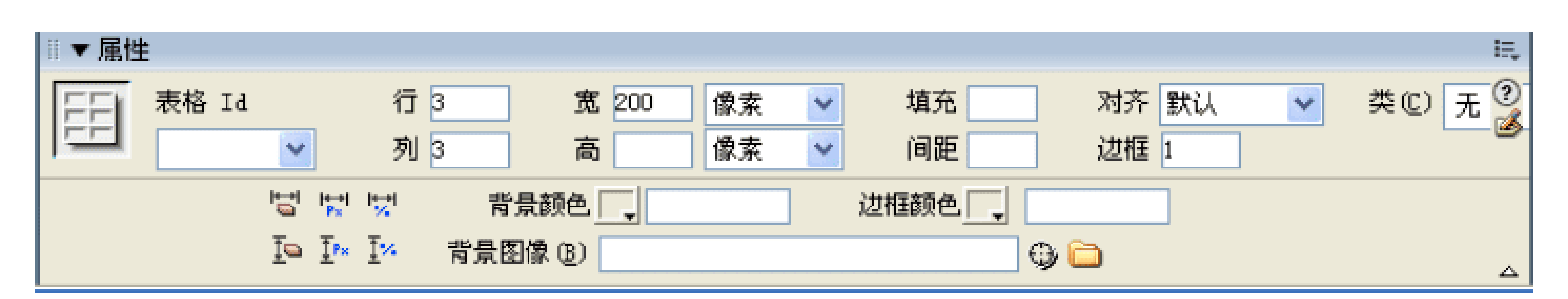


图 3.35　设置表格属性

在表格的【属性】面板中，大部分选项已在 3.3 节中介绍了，这里仅介绍【背景颜色】、【边框颜色】和【背景图像】等选项。

- 【背景颜色】：主要用于设置当前表格的背景颜色，用户可通过直接在对应的文本框中输入颜色的十六进制数值来设置，或通过单击按钮，在弹出的调色板中进行选择。

- 【边框颜色】：主要用于设置当前表格边框的颜色，其设置方法与【背景颜色】选项的设置方法相同。
- 【背景图像】：主要用于设置当前表格的背景图像，用户可直接在对应的文本框中输入背景图像的文件路径来设置，也可通过单击【文件夹】按钮，在弹出的【选择文件】对话框中选择图像文件。此外，设置背景图像还可通过按住【指向文件】按钮，拖动鼠标至【文件】面板中相应的图像文件来实现。

3.4.3　单元格设置

单元格是表格对象的一个基本元素，因此对单元格的设置是处理表格的一个基本操作。一般来说，对单元格的设置主要包括单元格属性的设置以及单元格的合并与拆分。

1. 单元格属性

将光标置于表格的单元格中，【属性】面板中的选项将变为设置单元格属性的选项，如图 3.36 所示。

图 3.36　设置单元格属性

在该【属性】面板中，上方选项用来设置单元格中的文本的属性，下方选项用来设置单元格的属性，这里仅对后者进行介绍。

- 【水平】：用于设置单元格中的文本的水平对齐方式，其选项包括【左对齐】、【居中对齐】和【右对齐】，其默认值为【左对齐】。
- 【垂直】：用于设置单元格中的文本的垂直对齐方式，其选项包括【顶端】、【居中】、【底部】、【基线】，其默认值为【居中】。
- 【宽】和【高】：分别用于设置单元格的宽度和高度，其单位默认为像素。也可通过输入百分比来设置当前单元格占整个表格宽度或高度的百分比。
- 【不换行】：用于设置单元格中的文本是否允许换行。如果选中此项，则当单元格中的内容超过单元格的宽度时，单元格将自动加宽以容纳所有内容。
- 【标题】：用于设置当前的单元格格式是否为表格标题单元格。默认情况下，表格标题单元格的内容将以粗体显示并且居中。
- 【背景】：用于设置单元格的背景图像，其操作与表格的背景图像设置相同。
- 【背景颜色】：用于设置单元格的背景颜色，其操作与表格的背景颜色设置相同。
- 【边框】：用于设置单元格的边框颜色，其操作与表格的边框颜色设置相同。

2. 合并单元格

合并单元格是指将所选择的多个单元格合并成一行、一列或一个矩形。一般来说，可以合并任意数目的相邻的单元格，以生成一个跨多行或多列的单元格。

合并单元格的操作相当简单，只需选择需要合并的相邻的多个单元格，单击【属性】

面板中的□按钮即可。或在选择多个单元格后，选择【修改】|【表格】|【合并单元格】命令。

单元格合并前后的效果如图 3.37 和图 3.38 所示。

3. 拆分单元格

拆分单元格是指将当前选择的单元格拆分成多个单元格。

将光标置于所要拆分的单元格中，单击【属性】面板中的按钮，此时将弹出【拆分单元格】对话框，如图 3.39 所示。

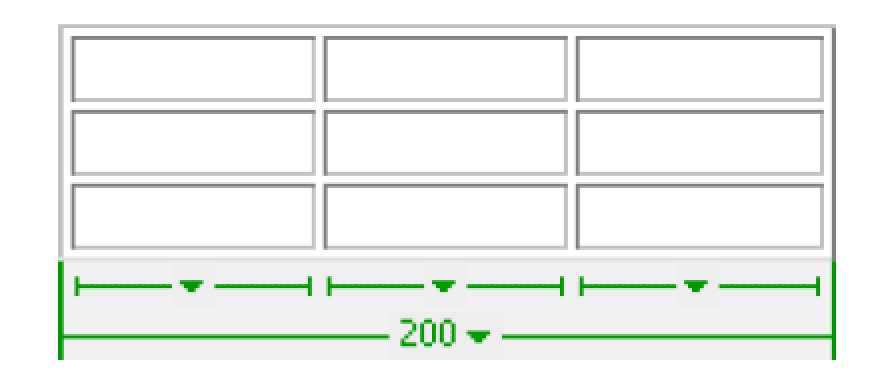


图 3.37　合并单元格之前

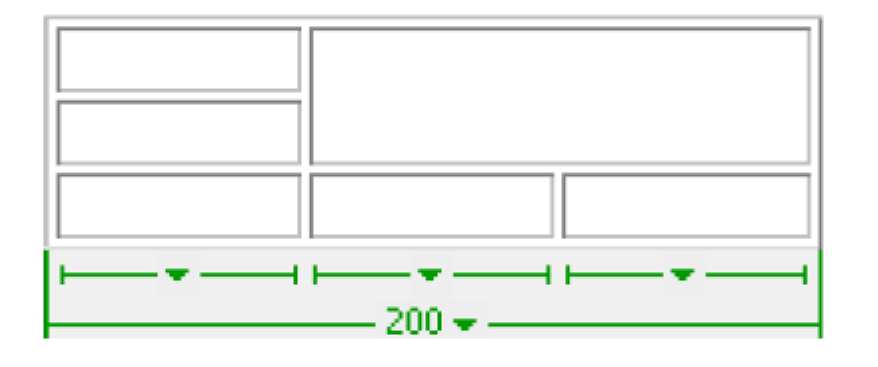


图 3.38　合并单元格之后

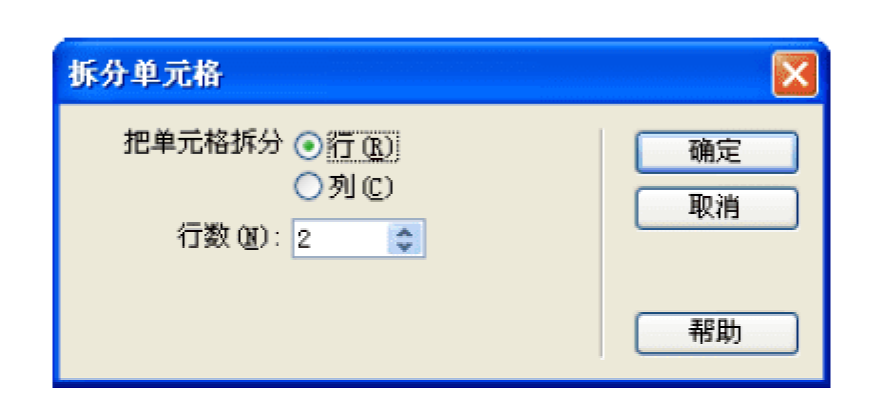


图 3.39　【拆分单元格】对话框

在此对话框中，首先需要设置的是将单元格拆分成多行还是多列，然后设置相应的行数或列数，最后单击【确定】按钮。

单元格拆分前后的效果如图 3.40 和图 3.41 所示。

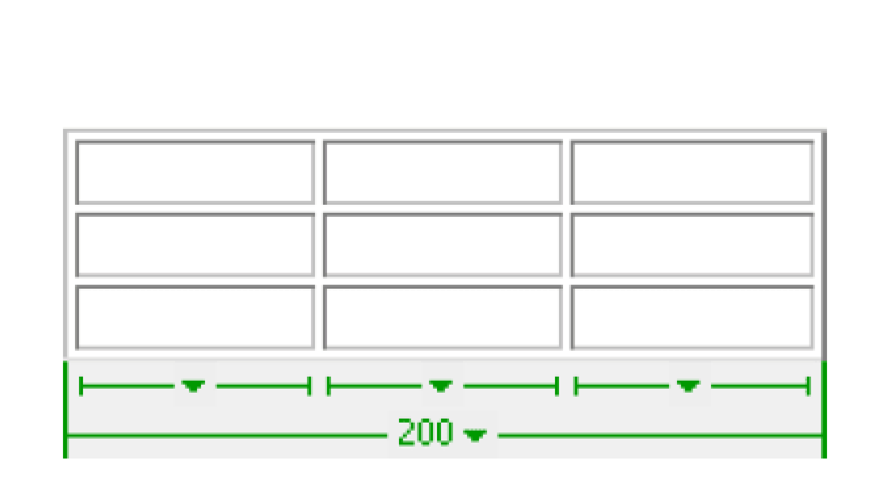


图 3.40　拆分单元格之前

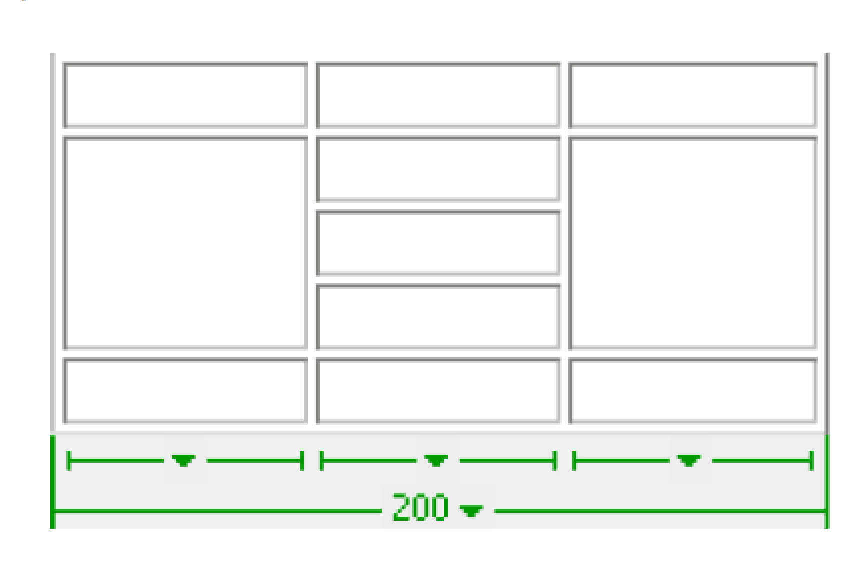


图 3.41　拆分单元格之后

3.5　典型实例：创建个人博客主页面

在本实例中，将通过一个博客主页面的制作来进一步掌握以上元素的使用。

(1) 启动 Dreamweaver 8，新建一个 HTML 页面，并将其命名为 Default.html。将视图切换至【设计】视图，单击【属性】面板中的【页面属性】按钮，在弹出的【页面属性】对话框中，设置页面的上边距为 0 像素，如图 3.42 所示。

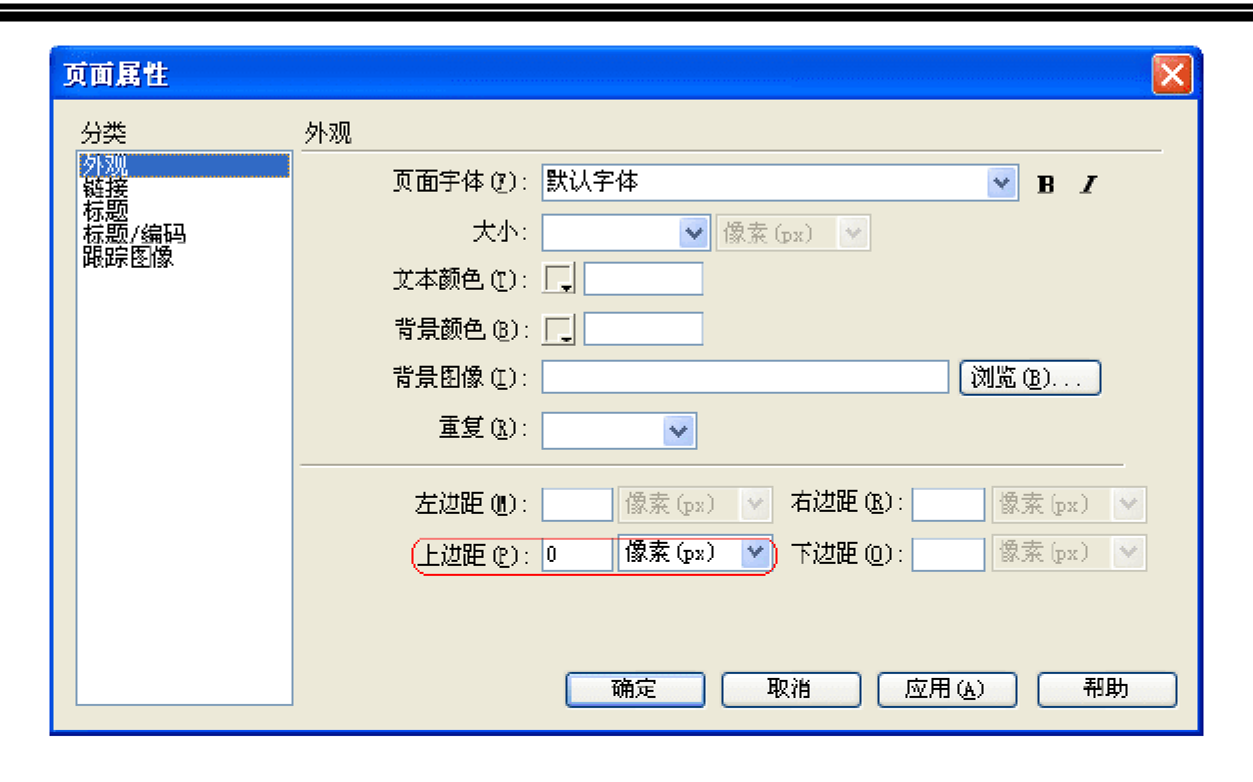


图 3.42　【页面属性】对话框

(2) 单击【确定】按钮，完成页面属性的设置。

(3) 单击【常用-插入】工具栏中的【表格】按钮，插入一个 2 行 2 列的表格 table1，并设置其宽度为 760 像素，对齐方式为居中对齐，边框为 0，如图 3.43 所示。该表格将用于控制页面的整体布局。

(4) 选择表格 table1 的第 1 行的两个单元格，单击【属性】面板中的按钮将其合并。将光标置于合并后的单元格中，单击【常用-插入】工具栏中的【图像】按钮，在页面中插入一个图像，该图像源文件为 Image 目录下的图像文件 Top.jpg。

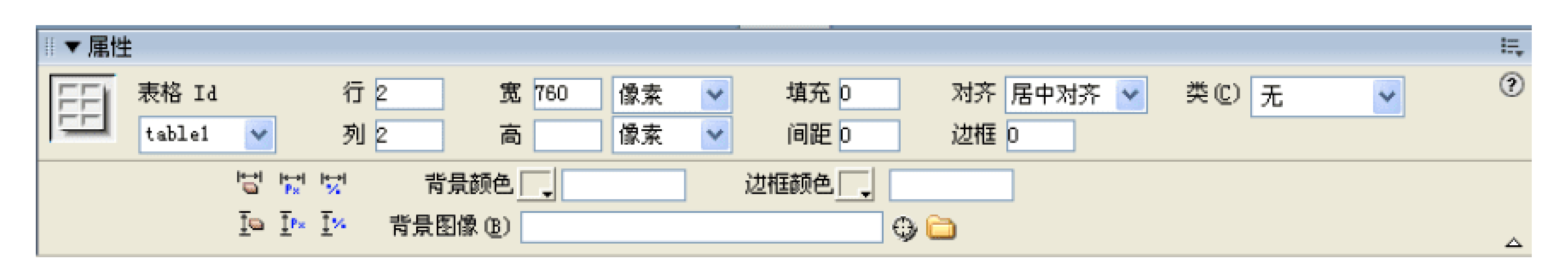


图 3.43　设置表格属性

(5) 选择表格 table1 的第 2 行的第 1 个单元格，设置其水平对齐方式为居中对齐，垂直对齐方式为顶端，宽度为 160 像素，高度为 350 像素，如图 3.44 所示。该单元格将用于显示个人形象及个人简介等信息。

图 3.44　设置单元格属性(1)

(6) 选择表格 table1 的第 2 行的第 2 个单元格，设置其水平对齐方式为右对齐，垂直对齐方式为顶端，宽度为 600 像素，高度为 350 像素，如图 3.45 所示。该单元格将用于显示个人日志信息。

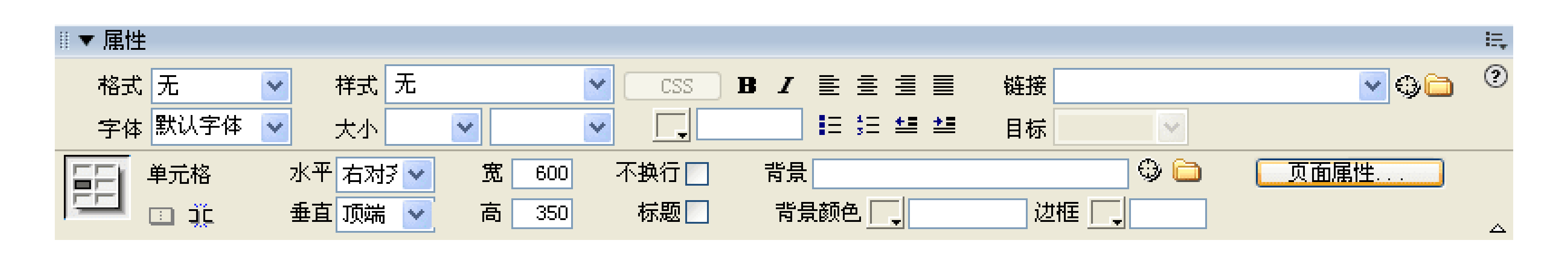


图 3.45　设置单元格属性(2)

(7) 将光标置于表格 table1 的第 2 行的第 1 个单元格中，单击【常用-插入】栏中的【表格】按钮，插入一个 2 行 1 列的表格 table2，并设置其宽度为 158 像素，间距为 1，边框为 0，背景颜色值为#333333，如图 3.46 所示。该表格将用于显示个人形象信息。请注意，这里表格的间距、边框及背景颜色属性的设置，均是为了创建一个细线边框表格。

提示： 创建细线边框表格是网页设计中经常需要的。其最简单的方法是，首先将表格的间距设置为 1，边框设置为 0，然后设置表格的背景颜色为所要创建的细线边框表格的边框颜色，最后再将各单元格的背景颜色设置为白色即可。

图 3.46　设置表格属性

(8) 将光标置于表格 table2 的第 1 个单元格中，并在【属性】面板中设置其背景颜色值为#ADC400，高度为 25，如图 3.47 所示。

图 3.47　设置单元格属性

(9) 在此单元格中输入文本“个人形象”，选择该文本，在【属性】面板中设置其字体颜色为白色，字体大小为 13 像素，并加粗显示。

(10) 将光标置于表格 table2 的第 2 个单元格中，并在【属性】面板中设置其背景颜色为白色，水平对齐方式为左对齐。然后，将视图切换至【代码】视图，添加该单元格的 Style 样式，如图 3.48 所示。

```
<td bgcolor="#FFFFFF" align=left style="padding-left:20px;padding-top:11px;font-size:13px;">
</td>
```

图 3.48　设置单元格的 Style 样式

(11) 单击【常用-插入】工具栏中的【图像】按钮，在此单元格中插入一个图像，其源文件为 Image 目录下的 MyPic.jpg。

(12) 选择表格 table2，将其复制，并在表格 table2 后粘贴，生成一个新的表格 table3。

(13) 将表格 table3 的第 1 个单元格中的文本更改为“个人简介”，然后将第 2 个单元格中的图像删除，并添加相应的个人简介信息，如图 3.49 所示。这里，对个人简介信息的文本格式设置不再叙述。

图 3.49　个人简介信息

(14) 将光标置于表格 table1 的第 2 行的第 2 个单元格中，单击【常用-插入】工具栏中的【表格】按钮，插入一个 2 行 1 列的表格 table4，并设置其宽度为 590 像素，高度为 100%，间距为 1，边框为 0，背景颜色值为#333333，对齐方式为居中对齐，如图 3.50 所示。

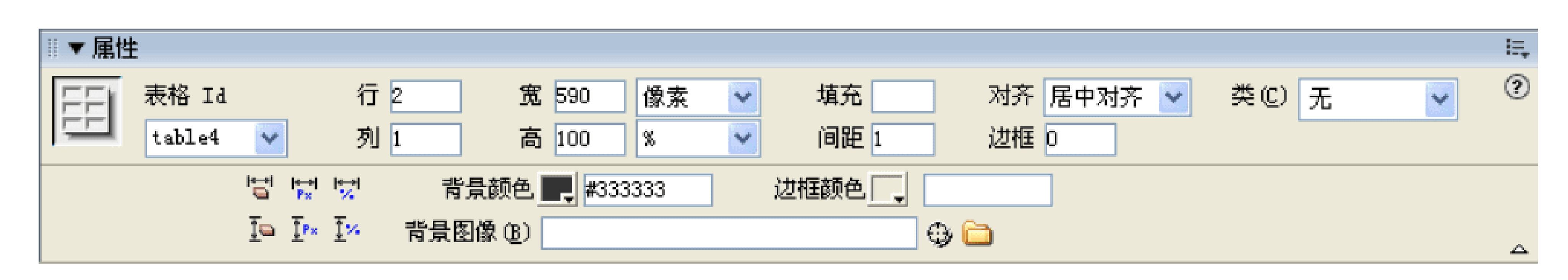


图 3.50　设置表格属性(1)

(15) 将光标置于表格 table4 的第 1 个单元格中，在【属性】面板中设置其背景颜色值为#ADC400，高度为 25，如图 3.51 所示。

图 3.51　设置表格属性(2)

(16) 在该单元格中输入文本“我的日志”，并设置其字体颜色为白色，字体大小为 13 像素，加粗显示。

(17) 在表格 table4 的第 2 个单元格中，输入相应的日志信息，如图 3.52 所示。

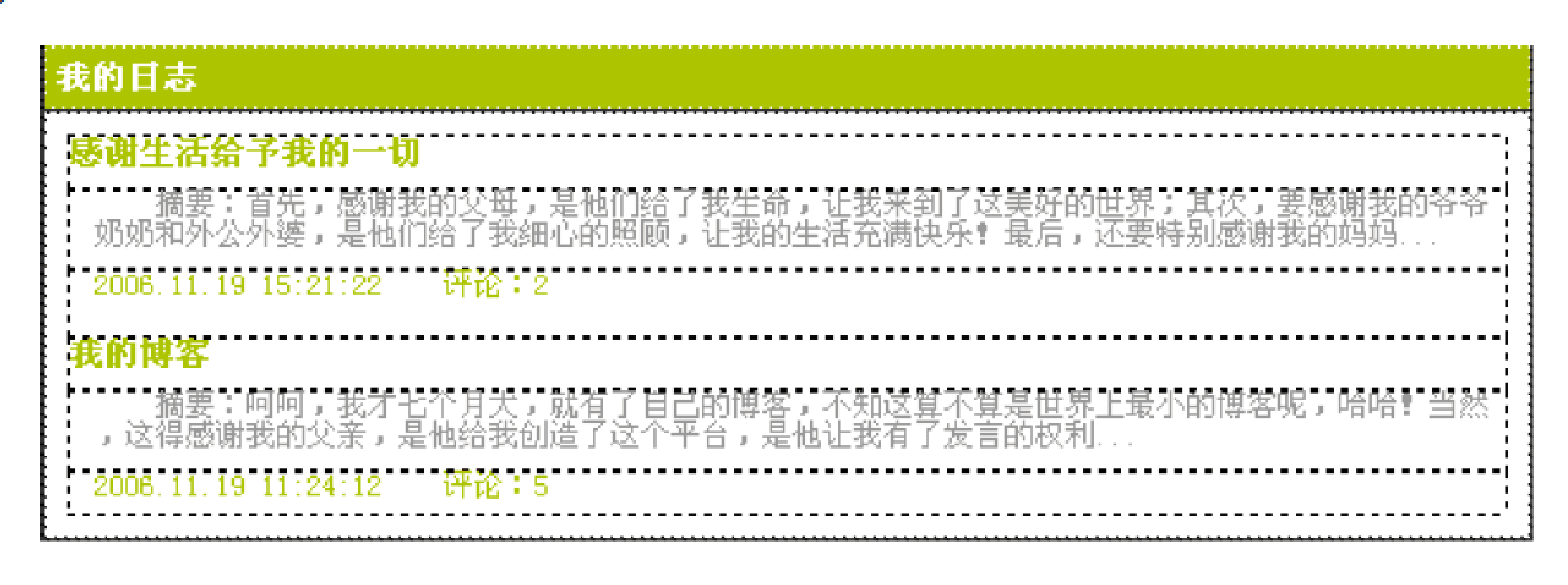


图 3.52　我的日志

需要说明的是，在正式的个人博客系统中，我的日志以及个人简介信息均是从数据库中取出的，而非直接输入的固定文本。这里为了说明页面的布局实现，采用了静态的文本来替代动态文本的显示。因此，这里没有对个人简介信息及我的日志内容的文本格式设置及具体布局作进一步的叙述。

至此，一个基本的个人博客主页面已经成型，页面预览效果如图 3.53 所示。

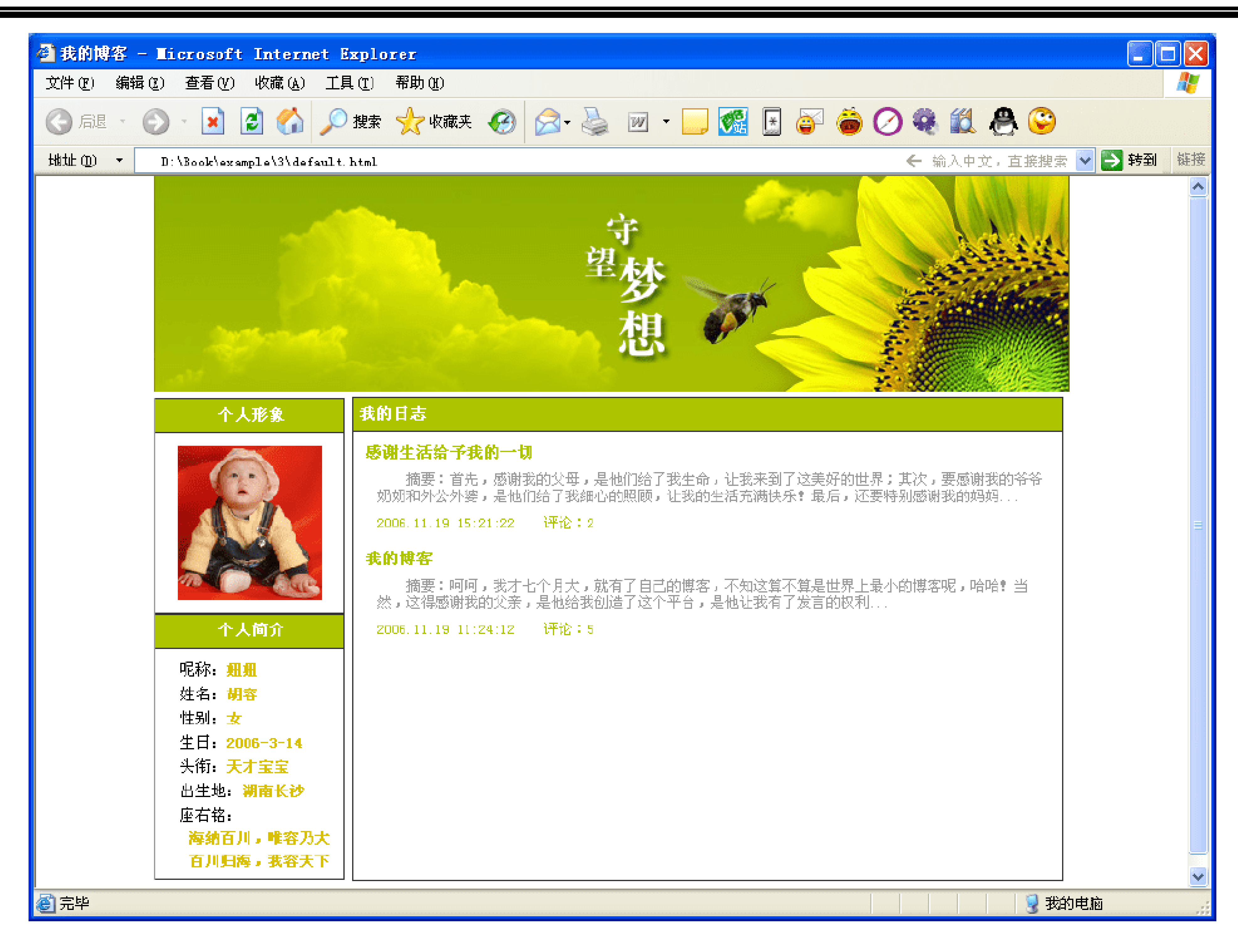


图 3.53　页面预览效果

3.6　习　　题

(1) 尝试创建一个细线边框表格，并将细线边框的颜色值设置为#0000FF。

(2) 在本章实例的“个人简介”一栏中，添加电子邮箱一项，并设置相应的电子邮件链接。

(3) 在本章实例中，使用一个较大的图像替换页面顶部的 Top.jpg 文件，并在该图像的【属性】面板中设置其【低解析度源】选项，即指定一个图像加载过程中的替代图像，并查看页面加载效果。

第 4 章　数据库的操作

在第 3 章中，详细地介绍了 Dreamweaver 中的各种基本操作。这一章，主要介绍数据库的基本概念及最基本的操作。

4.1　建立数据库

一直以来，数据库都是 Web 应用程序开发中的一个重要组成部分，对数据库的设计和操作也是 Web 应用程序设计和实现的一个重要环节。

4.1.1　数据库的概念

什么是数据库？打个比方，每个人都会有自己的地址簿，上面记录了各个亲朋好友的姓名、地址、联系电话等。这个实际上就是一个简单的“数据库”，它所记录的每个人的信息就是“数据库”中的数据。通过查询地址簿，可以知道如何联系上面的每个人；同时，也可以不断地往地址簿上添加更多朋友的联系方式。

简单地说，数据库就是按某种规则将数据组织起来的持久数据的集合，这些数据可用于一些特定的应用中。在数据库中，用户可以按照特定的方式来存储数据，同时一旦数据被存储至数据库中，就可以方便地查询其中的信息。此外，对于数据库中的信息，还可进行相应的添加、修改和删除。

目前，常用的数据库包括 Oracle、Microsoft SQL Server、Microsoft Access、Sybase、DB2 等，而在 Web 应用程序开发中，尤以 Microsoft SQL Server 数据库和 Microsoft Access 数据库应用最多。

在本书中的所有示例中，均使用 Microsoft Access 2000 数据库。本章将着重介绍 Microsoft Access 2000 数据库的各项基本操作。

Microsoft Access 是一个关系型数据库系统，它也是微软 Office 办公套件中的一个重要组成部分。与其他数据库相比，Microsoft Access 数据库更加简单易学，对于普通的计算机用户均可掌握并使用它。Microsoft Access 数据库的功能相当强大，完全可以应付一般的数据管理及处理需要。

提示：所谓关系型数据库，其基本特征是按照关系数据模型组织数据库，具体表现为以行和列的形式来存储数据，一系列的行和列就构成了数据表，一系列的数据表构成了数据库。关系型数据库以其结构简单、理论基础坚实、数据独立性高以及提供非过程性语言等优点而被认为是具有很大发展前景的一种数据库，Microsoft Access 数据库便是其中之一。

4.1.2　建立 Access 数据库

创建 Access 数据库的方法有两种：一种方法是创建一个空的数据库，即一个没有数据表、查询、窗体和报表等内容的数据库；另一种方法是通过 Access 中提供的数据库模板来创建数据库，即通过向导中的选项来设计窗体、查询和报表等，从而建立一个完整的数据库。

一般来说，更多的用户需要创建一个空的数据库，然后在其中添加新的数据表。这里，以创建空的数据库为例，来介绍如何建立 Access 数据库，其具体步骤如下。

(1) 运行 Microsoft Access，此时将弹出如图 4.1 所示的对话框，提示用户目前所要执行的操作，其中包括【空 Access 数据库】、【Access 数据库向导、数据页和项目】和【打开已有文件】3 个选项。

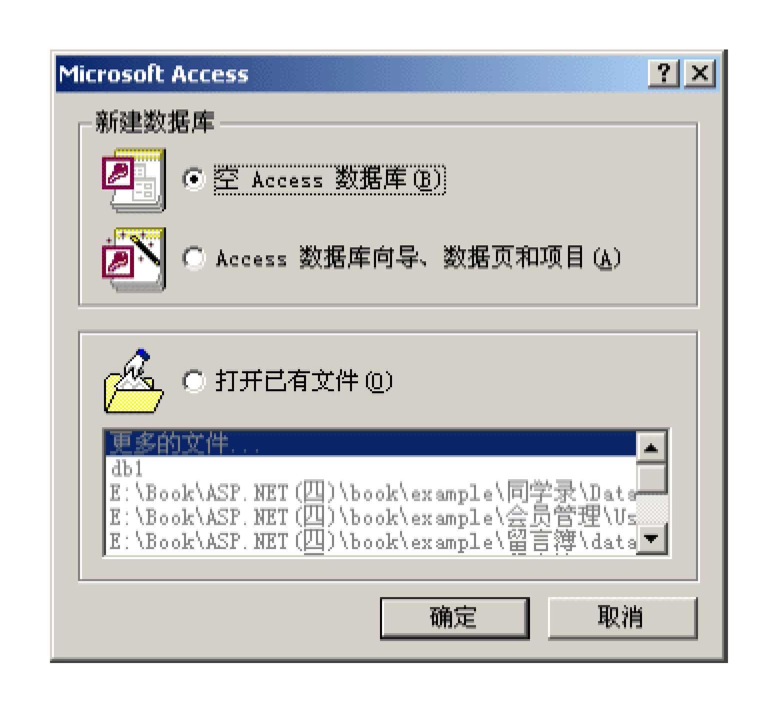


图 4.1　Microsoft Access 对话框

如果 Microsoft Access 已经运行，用户也可选择【文件】|【新建】命令，此时将弹出如图 4.2 所示的【新建】对话框。

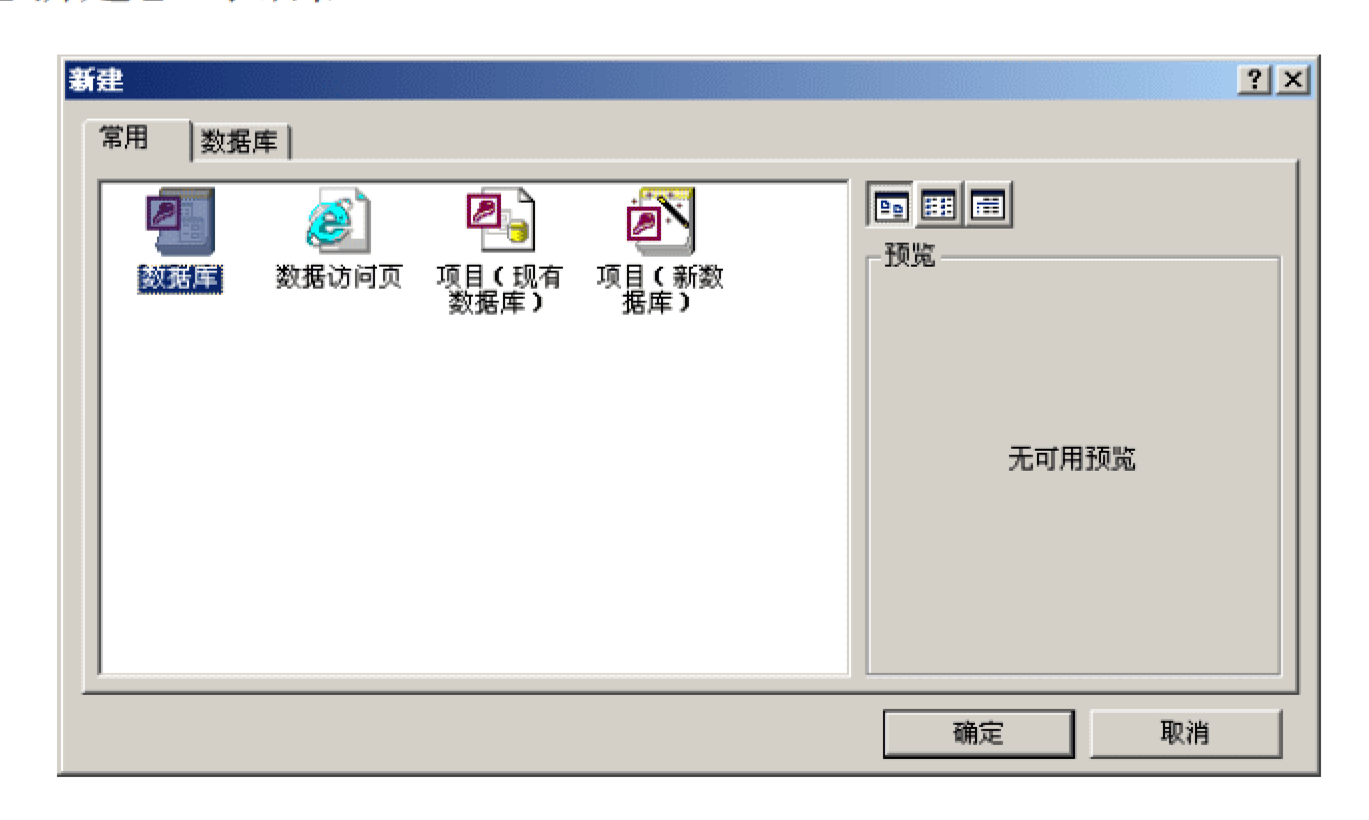


图 4.2　【新建】对话框

(2) 在 Microsoft Access 对话框中选择新建数据库中的【空 Access 数据库】选项，或在【新建】对话框中选择【数据库】选项，然后单击【确定】按钮，将弹出如图 4.3 所示的【文件新建数据库】对话框。

(3) 在【文件名】文本框中输入所要新建的数据库名(如 MyDB)，并选择新建数据库所要保存的路径，然后单击【创建】按钮，即可创建一个名为 MyDB 的空 Access 数据库，

并显示如图 4.4 所示的窗口。

图 4.3　【文件新建数据库】对话框

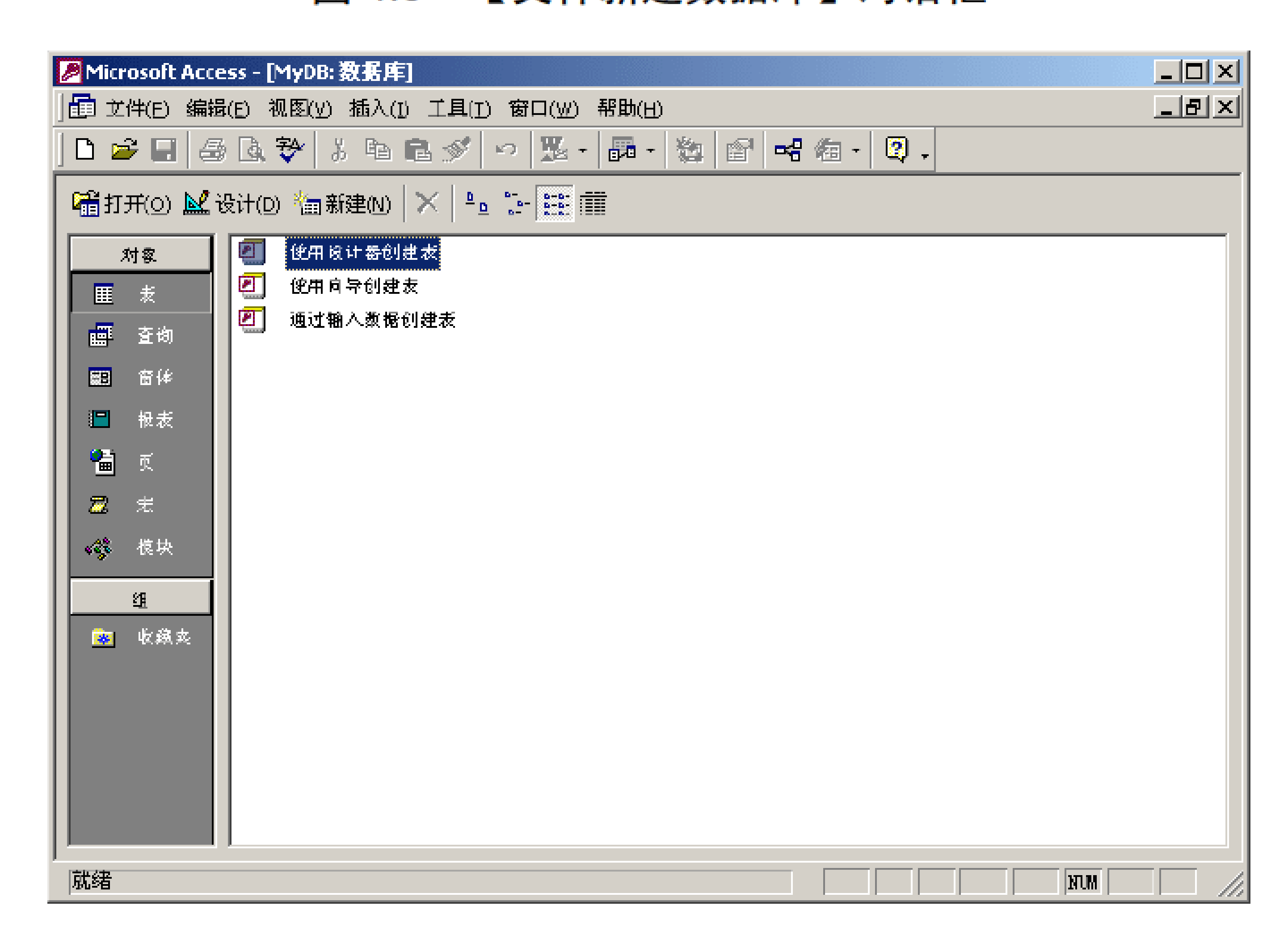


图 4.4　【MyDB：数据库】窗口

从此窗口中也可看出，所创建的 MyDB 数据库中没有任何数据表，而仅有三个用于创建数据表的相关向导。

4.1.3　建立数据表

数据表是 Access 数据库中最基本的对象，也是数据库中所有数据的载体。数据库中的其他对象，如查询、窗体、报表等，也是将表中的信息以各种形式表现出来，方便用户使用。

数据表可以分为两部分：表结构和表数据，而在本小节中所要叙述的是创建数据表的表结构。表结构是指数据表的框架，其基本要素是字段。每个字段都有自己的属性，主要包括如下。

1. 字段名称

字段的名称，用于标识表中的一列。对于同一个表中的字段名称，具有惟一性，即不允许两个同名的字段存在。

2. 数据类型

根据关系数据库理论，一个数据表中的同一列数据必须具有共同的数据特征，这就是数据类型。在 Access 中，所提供的数据类型包括：文本、数字、备注、日期/时间、货币、自动编号、是/否、OLE 对象、超级链接等，其说明见表 4.1。

表 4.1　Access 中所提供的数据类型及说明

数据类型	使用说明	字段大小
文本	用于存储文本信息，如姓名、地址等；或任何不用作计算的数字的数据，也可以是文本与数字的组合	最长为 255 个字符
数字	用于存储需要进行数学运算的数字数据	由“字段大小”属性中所设置的数字类型所决定
备注	用于存储多于 255 个字符的文本数据	最长为 64 000 个字符
日期/时间	用于存储日期及时间数据	8 个字节
货币	用于存储类似于金额的数字数据；可避免在计算时发生四舍五入的情况；可精确到小数点左边 15 位及右边 4 位	8 个字节
自动编号	此字段可在添加记录时自动插入依次递增的惟一序号(每次加 1)或随机生成的编号	4 个字节
是/否	此字段仅记录用户所选择的值对中的一种取值，如是/否、真/假、开/关等；可通过字段中的“格式”属性来确定值对的类型	1 个字节
OLE 对象	用于存储诸如 Microsoft Word 或 Microsoft Excel 文档、图片、声音的数据以及在别的程序中所创建的其他类型的二进制数据	字段大小受磁盘空间限制，最大为 1GB
超级链接	用于存储超级链接数据	最长为 64 000 个字符
查阅向导	用于存储从已有的表或查询中所查阅到的值，或者是创建字段时所输入的一组固定值的集合	通常为 4 个字节

3. 字段大小

字段大小，是指数据表中的一列所能容纳的最大字符个数。默认情况下，字段大小以字节为单位。

4. 其他属性

除以上属性外，字段还具有其他属性，如格式、默认值、有效性规则、有效性文本、必填字段、索引等。对于不同数据类型的字段，其所具有的相关属性将会有所不同。

在对数据表有所了解之后，下面来看如何建立数据表。

在如图 4.4 所示的数据库窗口中可以看到，创建数据表有 3 种途径：使用设计器创建表、使用向导创建表和通过输入数据创建表。这里，主要介绍如何通过设计器和通过输入数据两种方法来创建一个记录地址簿信息的数据表。

例：通过设计器来创建数据表。

通过设计器来创建数据表的具体步骤如下。

(1) 双击【MyDB：数据库】窗口中的【使用设计器创建表】选项，将弹出数据表的设计窗口，如图 4.5 所示。

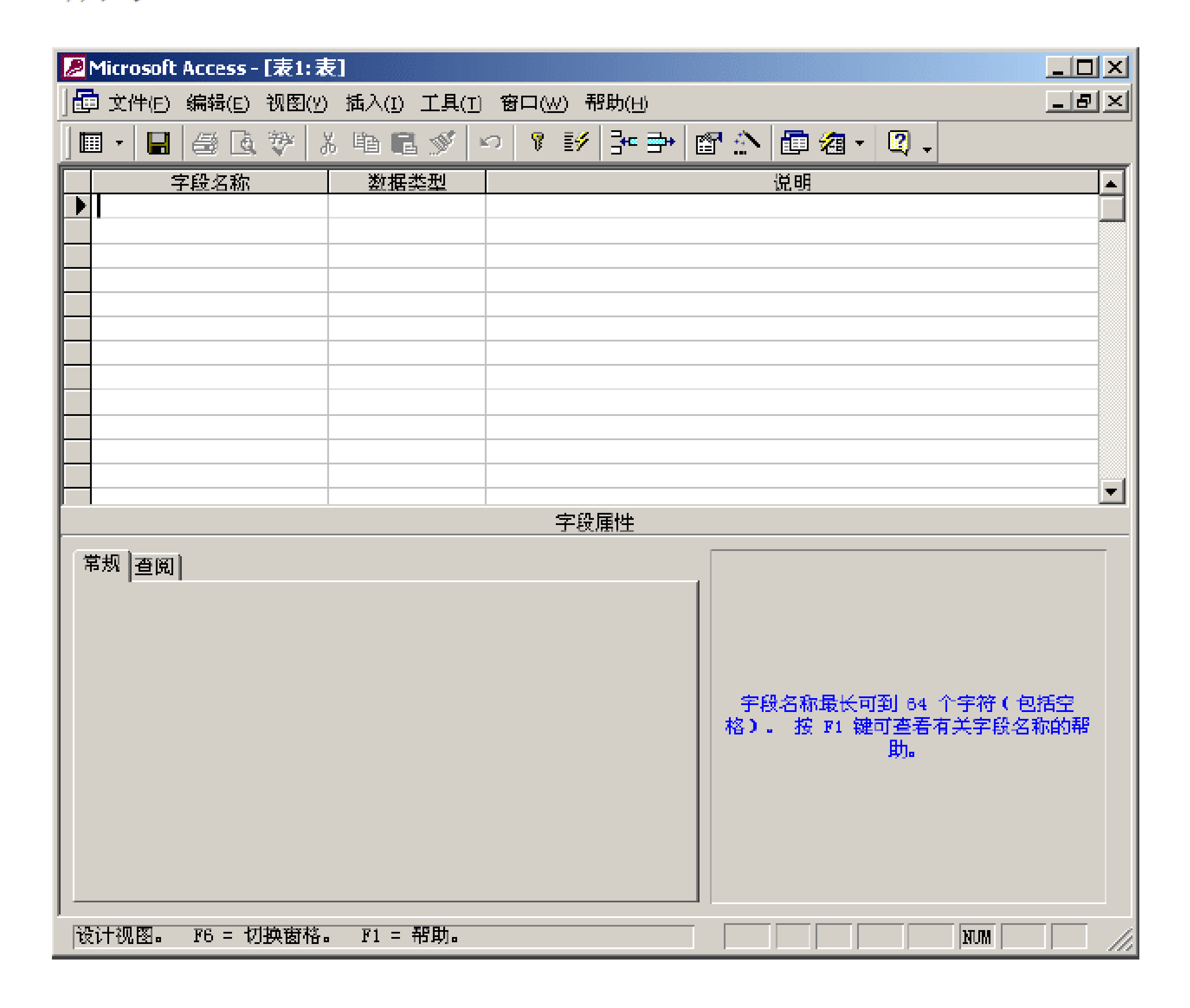


图 4.5　表设计窗口

(2) 在【字段名称】列中，输入第一个字段的名字 Name(表示姓名)，然后按 Enter 键，此时在【数据类型】列中将会显示一个类型选择按钮；单击该按钮，在弹出的下拉列表中选择【文本】选项，设置 Name 字段的类型为文本型，如图 4.6 所示。

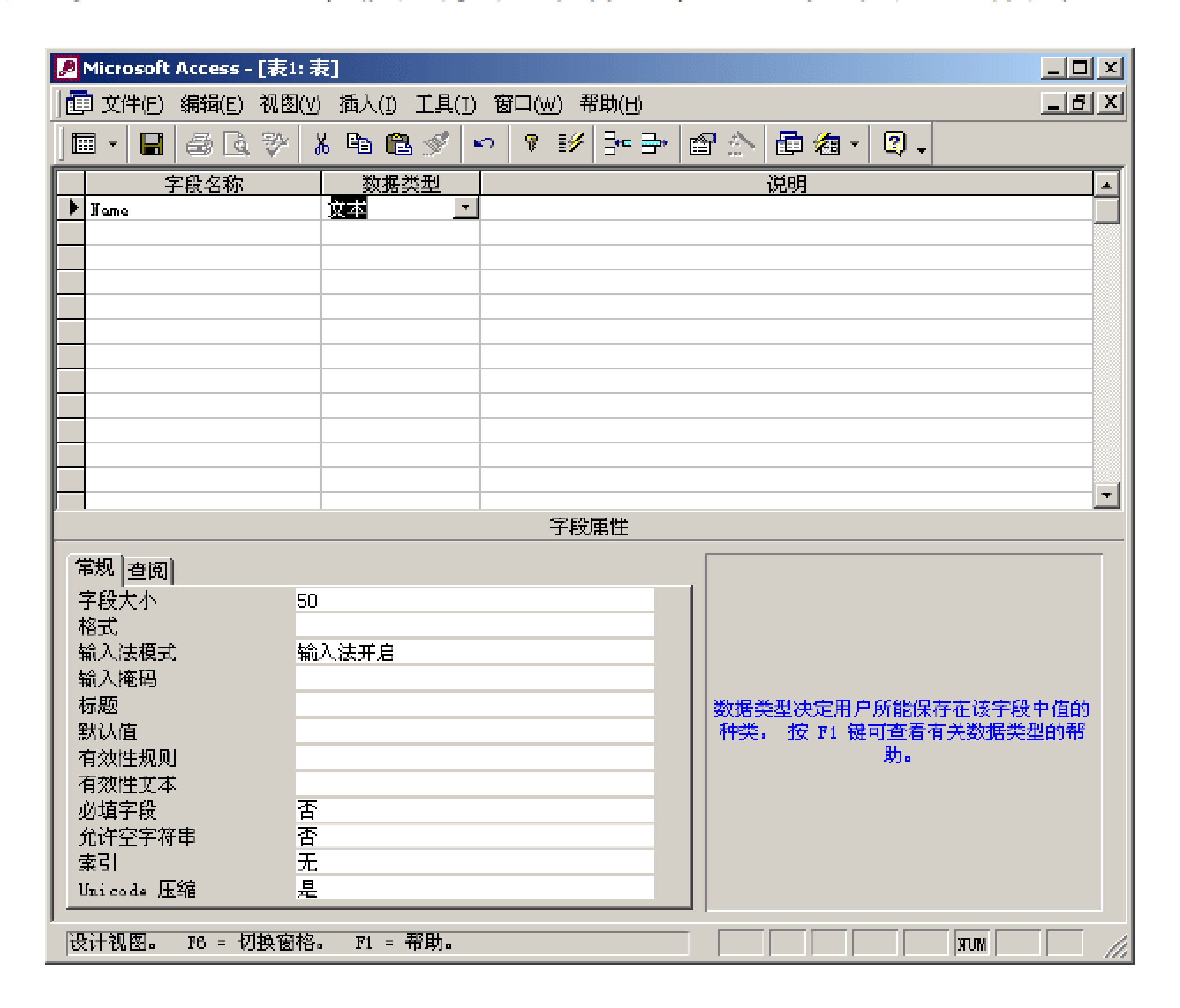


图 4.6　添加字段 Name

(3) 此时，在【字段属性】域的【常规】选项卡中，显示了该字段的其他属性，包括字段大小、格式、输入法模式、默认值、必填字段、索引等。这里，将字段 Name 的大小设置为 20，【必填字段】选项设置为【是】(即该字段为必须输入的字段)，【允许空字符

串】选项设置为【否】(即不允许输入为空)，如图 4.7 所示。

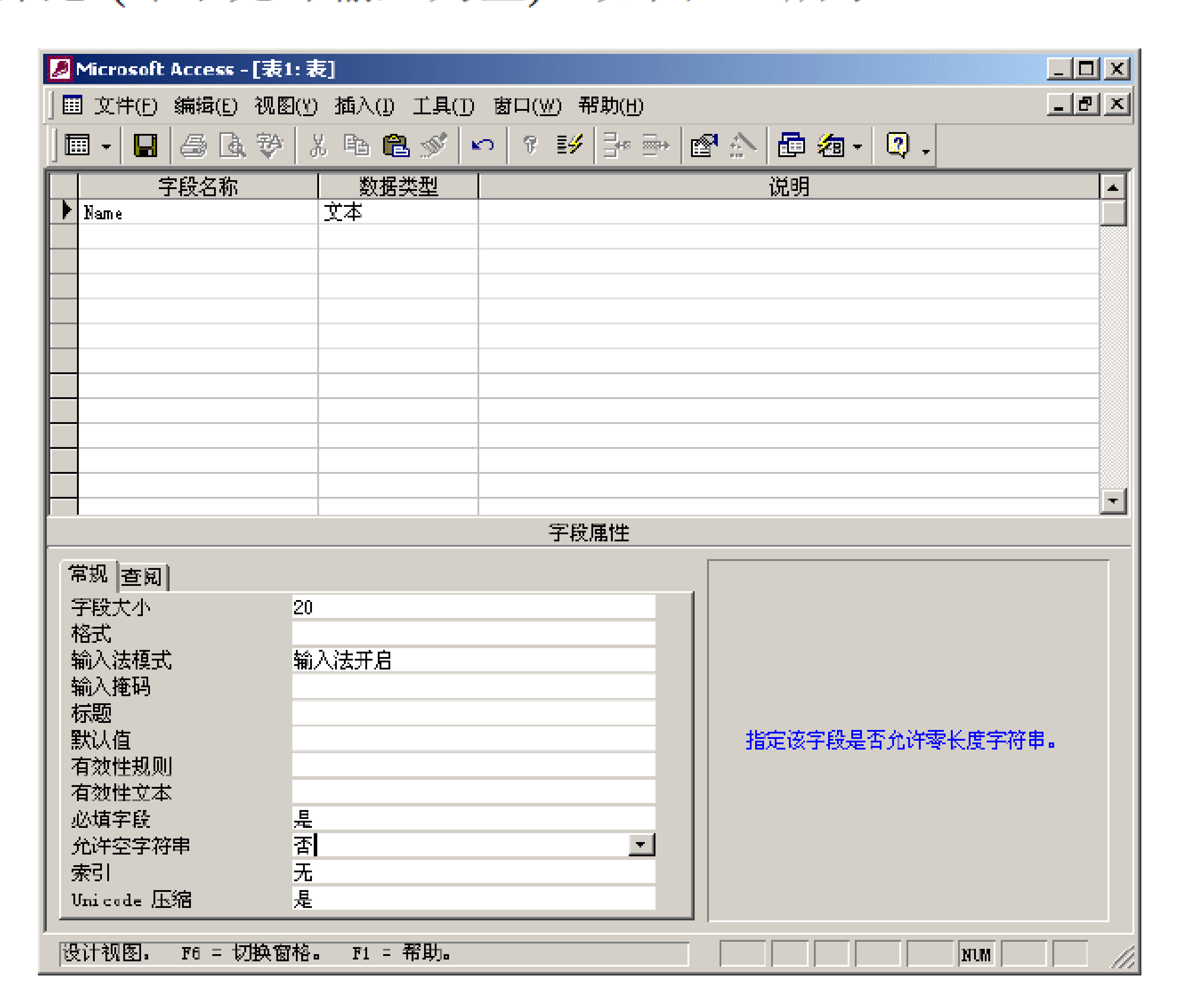


图 4.7　设置字段属性

(4) 重复步骤(2)和(3)，依次添加其他字段，并设置其字段名称、数据类型及字段大小等相关属性，如图 4.8 所示。

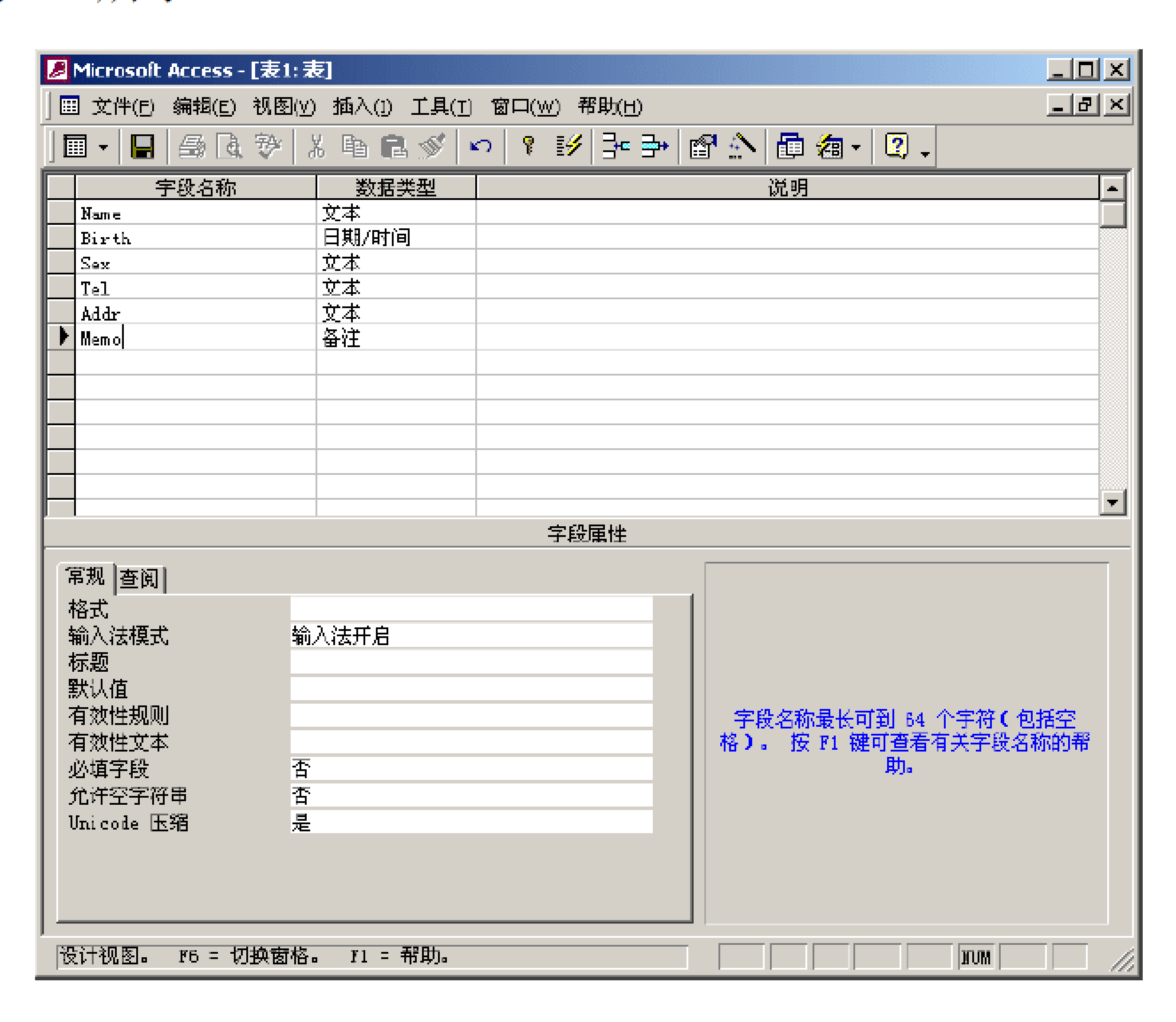


图 4.8　依次添加其他字段

在以上添加的字段中，Birth 表示出生日期，其数据类型为日期/时间型；Sex 表示性别，其数据类型为文本型；Tel 表示联系电话，其数据类型为文本型；Addr 表示联系地址，其数据类型为文本型；Memo 表示个人备注信息，其数据类型为备注型。

(5) 字段添加完成后，单击按钮(【保存】按钮)，将弹出【另存为】对话框，要求输入该数据表的名称，这里将该表命名为“地址簿”，如图 4.9 所示。

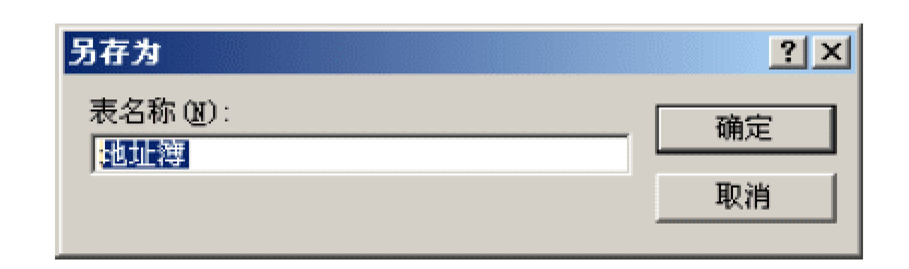


图 4.9　【另存为】对话框

提示： 一般情况下，在实际应用中，数据表的命名尽量不要使用中文，最好使用英文。因为在程序设计时，英文的输入比中文的输入要快，这会给用户带来一定的便利；此外，在 Web 应用中，也可避免由于某些浏览器对中文的不支持而带来一些问题。此处，仅仅是便于用户明白该表的用途，因此采用了中文命名。

(6)　单击【确定】按钮，系统将弹出【尚未定义主键】对话框，如图 4.10 所示。

图 4.10　【尚未定义主键】对话框

Access 对于所创建的每一个表都要求定义一个惟一的主关键字段，即主键。所谓主键，是数据表中每一条记录的惟一标识。为确保惟一性，Access 禁止在主键字段中输入重复值或 NULL 值。

提示： 主键字段不一定是一个字段，它也可以是多个字段的组合，即这多个字段的值在数据表中是惟一的。

对于如图 4.10 所示的对话框，如果单击【是】按钮，系统将自动创建一个名为 ID 的主键字段，其数据类型为“自动编号”；如果单击【否】按钮，系统将取消保存操作，返回表的设计页面，等待用户自定义一个主键；如果单击【取消】按钮，系统将取消主键的创建并保存该表。在这里，单击【取消】按钮。

至此，数据表的创建工作全部完成，用户可以在数据表中添加数据了，此步操作将在 4.2 节介绍。

例：通过输入数据创建数据表。

通过输入数据创建数据表，也就是用户首先输入数据，然后 Access 根据用户输入的数据来创建一个数据表。

双击数据库窗口中的【通过输入数据创建表】选项，系统将弹出数据表的数据输入窗口，如图 4.11 所示。

默认情况下，数据输入窗口提供了 10 个字段，字段名分别为“字段 1”、“字段 2”、……、“字段 10”。用户可以直接在该窗口中输入数据，输入了多少个字段的数据，所创建的表就有多少个字段。各字段的数据类型由 Access 根据用户所输入的数据自动进行判断，例如，用户在某列输入了数值，则该列被认为是数字类型；在某列输入了字符，则该列将被认为是文本类型。

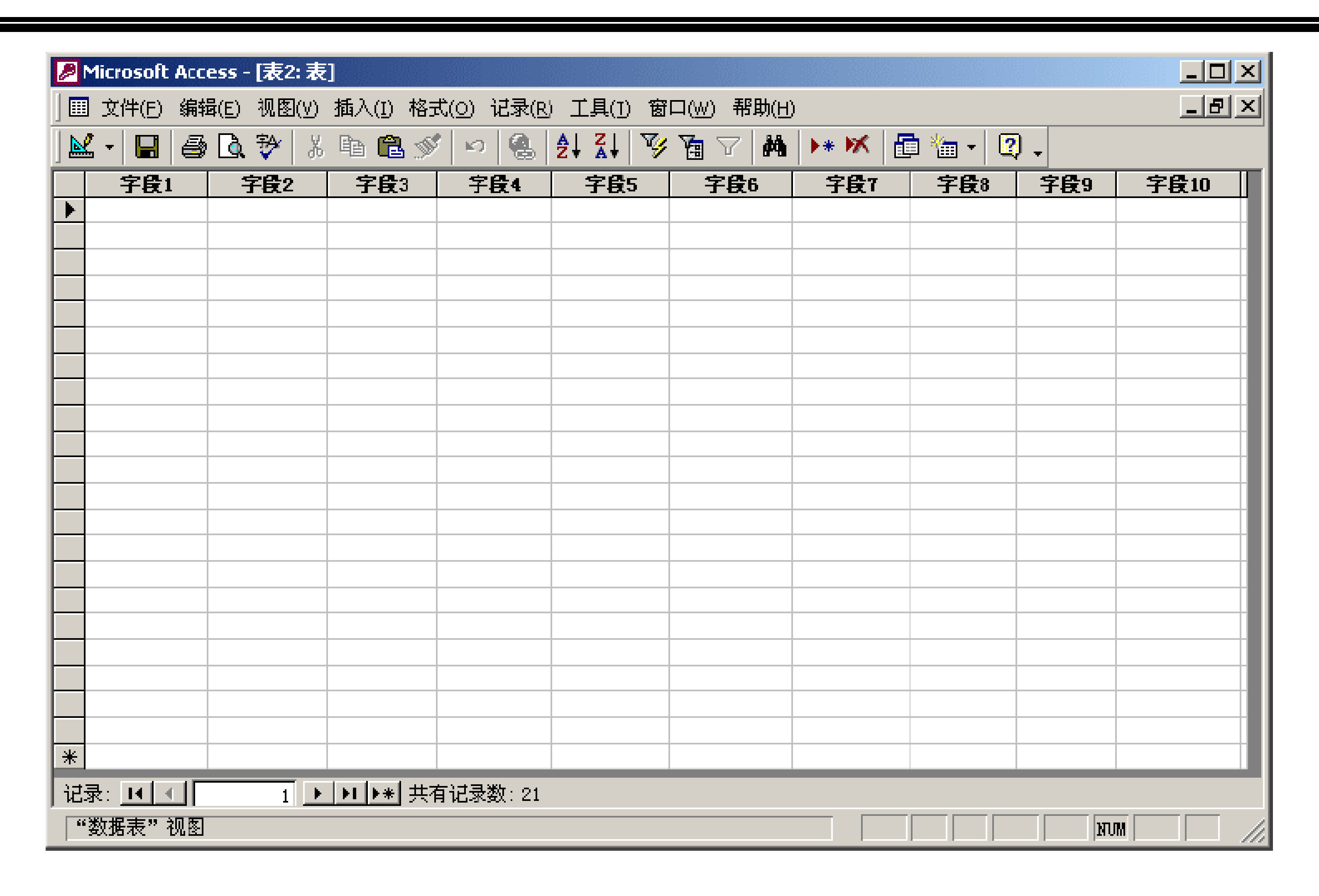


图 4.11　数据表的数据输入窗口

此方法可以在输入数据的同时创建新表，给用户带来一定的便利。但是，在实际操作中，由于表的字段名是默认的，在输入数据的同时不能修改表的字段名。因此，用户往往还需要在创建该表后，对表进行编辑，把表的字段名修改为自己所需要的字段名。

通常情况下，建议采用表的设计器来创建数据表。

4.1.4　数据库的打开与关闭

打开 Access 数据库的方法有三种。

(1) 运行 Microsoft Access，此时将弹出如图 4.12 所示的对话框。选择【打开已有文件】单选按钮，在下面的列表框中显示了最近打开过的 Access 数据库。如果要打开的数据库已在其中，则可选中该数据库的名称并单击【确定】按钮，或直接双击。

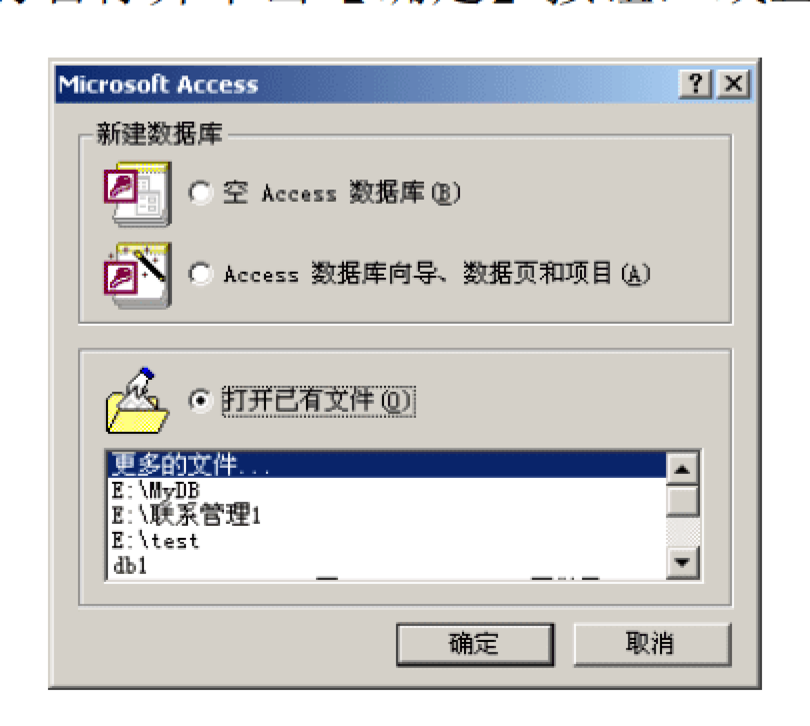


图 4.12　Microsoft Access 对话框

如果要打开的数据库不在列表中，则可双击【更多的文件】选项，此时将弹出【打开】对话框，如图 4.13 所示。

通过浏览磁盘文件夹，找到要打开的数据库，单击【打开】按钮。

(2) 如果 Microsoft Access 已经运行，此时可选择【文件】|【打开】命令，在弹出的【打开】对话框中，选择要打开的数据库文件。

(3) 在【资源管理器】或【我的电脑】中找到要打开的 Access 数据库文件(以.mdb 为

后缀名)，双击该文件，即可打开该数据库。

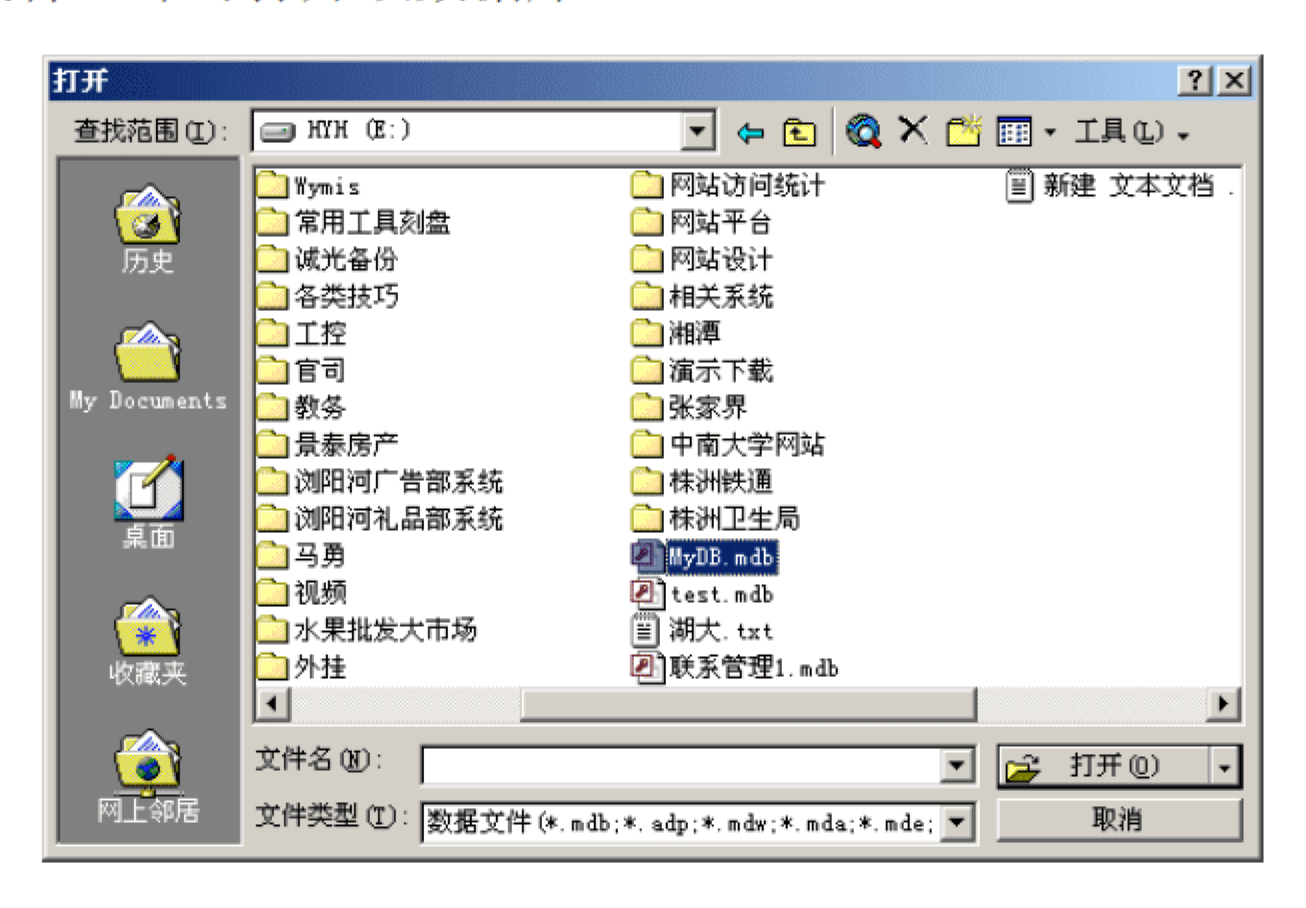


图 4.13　【打开】对话框

关闭 Access 数据库的方法也有以下 3 种。

(1) 选择【文件】|【关闭】或【退出】命令，均可关闭当前打开的数据库。前者，仅关闭数据库，但不关闭 Microsoft Access；而后者在关闭数据库的同时，也将关闭 Microsoft Access。

(2) 直接单击数据库窗口中的标题栏右侧的×按钮，关闭当前打开的数据库。

(3) 由于在一个 Microsoft Access 窗口中，只能打开一个 Access 数据库，因此可以通过打开另一个数据库的方法来关闭当前数据库。

4.2　数据库字段的操作

在实际运用中，对于创建的数据库表，常常由于各种原因需要对其表结构进行修改。接下来，介绍在 Access 中对数据库字段的一些常用操作。

4.2.1　主键字段设定

在创建名为“地址簿”的数据表时，曾提到主键字段是表中记录的标识性字段。而在前面创建数据表的步骤(6)中，单击的是【取消】按钮，也就是没有为该表创建主键。现在为该表设定主键。

在【MyDB：数据库】窗口中选择【地址簿】，然后单击【设计】按钮，或右击该表，从弹出的快捷菜单中选择【设计视图】命令，即可进入数据表的设计窗口，如图 4.14 所示。

选择 Name 字段，然后单击按钮，即可将 Name 字段设置为“地址簿”表的主键字段。此时，在 Name 字段的标头将会显示图标，表示该字段为主键。

注意：这里将保存“姓名”的字段设置为表的主键。

事实上，作为主键的字段其数据是不允许为空的，而作为姓名，很有可能会出现同名的情况。正是出于这种考虑，在 Access 中允许将多个字段同时设置为主键，只要这多个字段的数据不完全一样，也就不违反主键规则。因此，可以将 Name 字段和 Birth 字段同时设

置为主键，毕竟同一天出生且姓名完全相同的人很少。

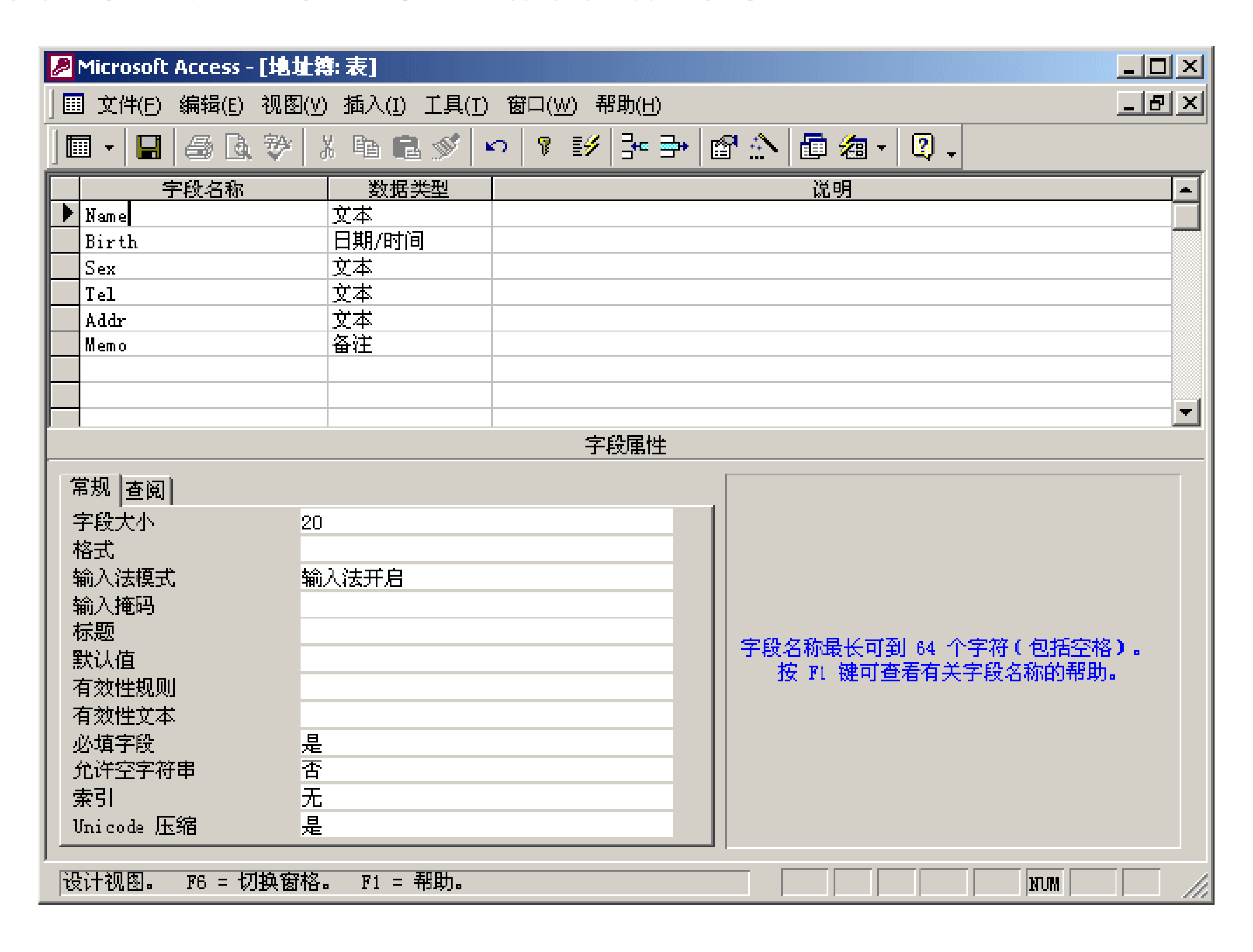


图 4.14　数据表的设计窗口

将多字段设置为主键的具体操作如下。

(1) 取消原有的主键设置。选择 Name 字段，然后单击按钮，此时该字段标头上的标识将会消失，表示已经取消了该字段的主键设置。

(2) 选择 Name 字段，按住 Ctrl 键，选择字段 Birth，单击按钮，即可完成主键的设置。此时，在 Name 字段和 Birth 字段的标头，均显示了标识，如图 4.15 所示。

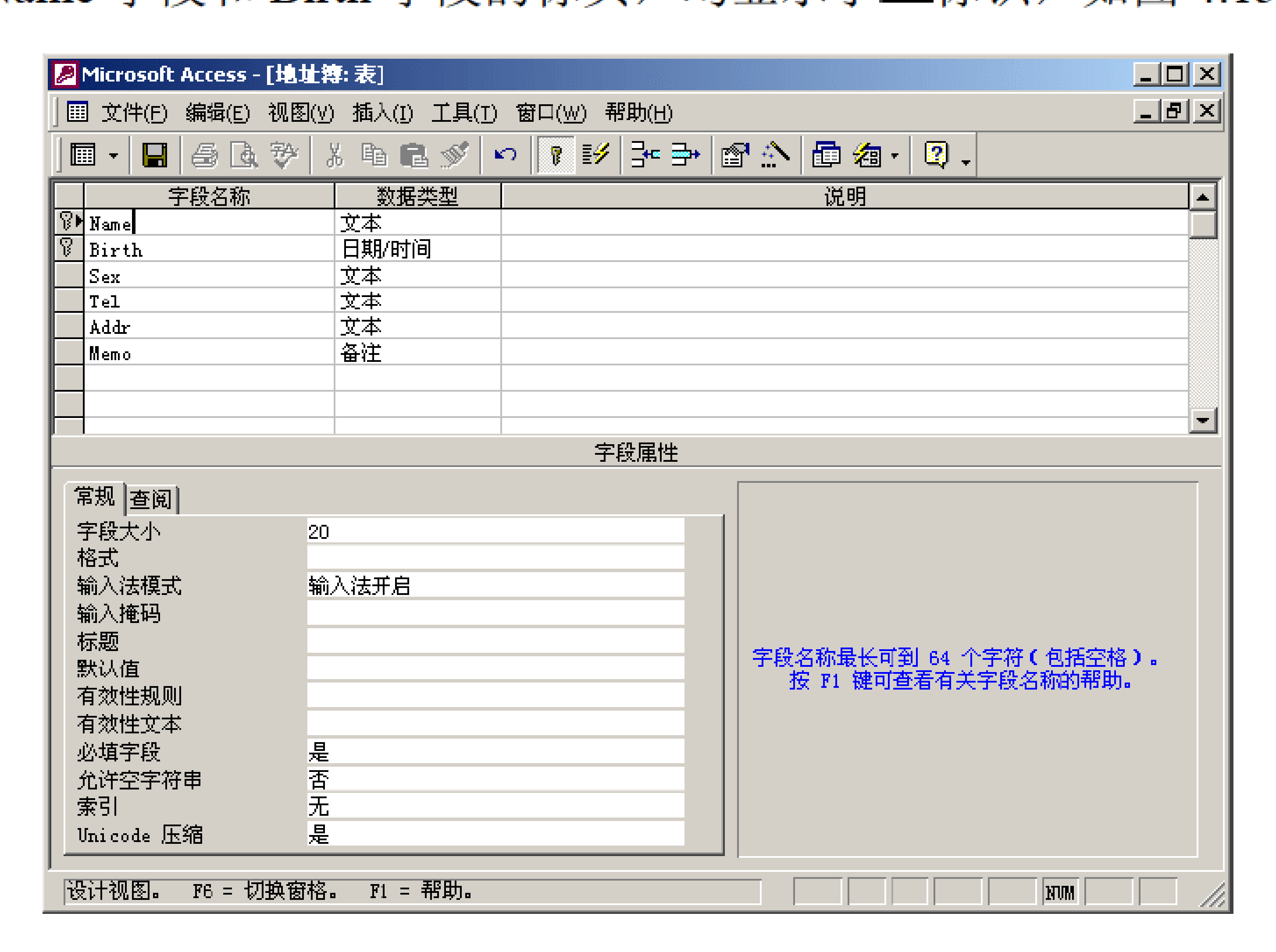


图 4.15　设置主键

设置 Name 字段和 Birth 字段同为主键，意味着这两个字段的数据均不能为空，也不能出现重复的情况，这是在编程应用中必须注意的。

4.2.2　添加字段

对已创建的数据表，可以添加新的字段，其操作如下。

选择要增加字段所在的行并右击，在弹出的快捷菜单中选择【插入行】命令。此时，将会在所选择的行处插入一个空行。然后，在空行中输入新字段的名称并设置数据类型等相关属性。

如果要在表的最后新增一个字段，则直接在表的最后一个字段的下一行输入要新字段的名称并设置相关属性即可。

操作完成后，单击按钮，保存所做的修改。

4.2.3　删除字段

删除字段的操作较为简单，其具体操作如下。

首先，选择要删除的字段并右击，在弹出的快捷菜单中选择【删除行】命令，或者直接按下 Delete 键，即可从数据表中删除该字段。

操作完成后，单击按钮，保存所做的修改。

4.2.4　调整顺序

如果需要调整字段的顺序，可选择需要移动位置的字段，然后按住鼠标并拖动鼠标至希望移至的位置再松开。此时，所选择的行将被移动到指定的位置上。

操作完成后，单击按钮，保存所做的修改。

4.2.5　设置字段必填属性

在字段的常规属性中，【必填字段】与【允许空字符串】是两个非常重要的属性，很多初学者常常无法分清这两个属性的区别。

所谓必填字段，是指字段中的数据必须有值。如果该属性设置为【是】，则表示该字段中必须有内容，而不能为 NULL(请注意，NULL 不等于空字符串)。

允许空字符串，表示该字段中的数据是否允许为空的字符串(如空格)，该属性在编写程序时需要特别注意。如果该字段不允许为空字符串，则在添加数据时必须保证该字段的数据不能为空。

4.3　数据的操作

前面介绍了数据库字段的操作方法，接下来介绍如何对数据表中的数据进行操作，包括数据的添加、删除和修改。

4.3.1　数据表中记录的添加

创建了数据表，便可以向数据表中添加数据。要知道，数据表只是一种载体，而数据才是用户真正需要的。

向数据表中添加数据，有两种方法：一种是直接打开表的浏览窗口，添加数据；另一种则是在查询窗口中通过 SQL 语句向表中添加数据。这里，主要介绍第一种方法。

在数据库窗口中，双击要添加数据的数据表，即可打开该表的浏览窗口，如图 4.16

所示。

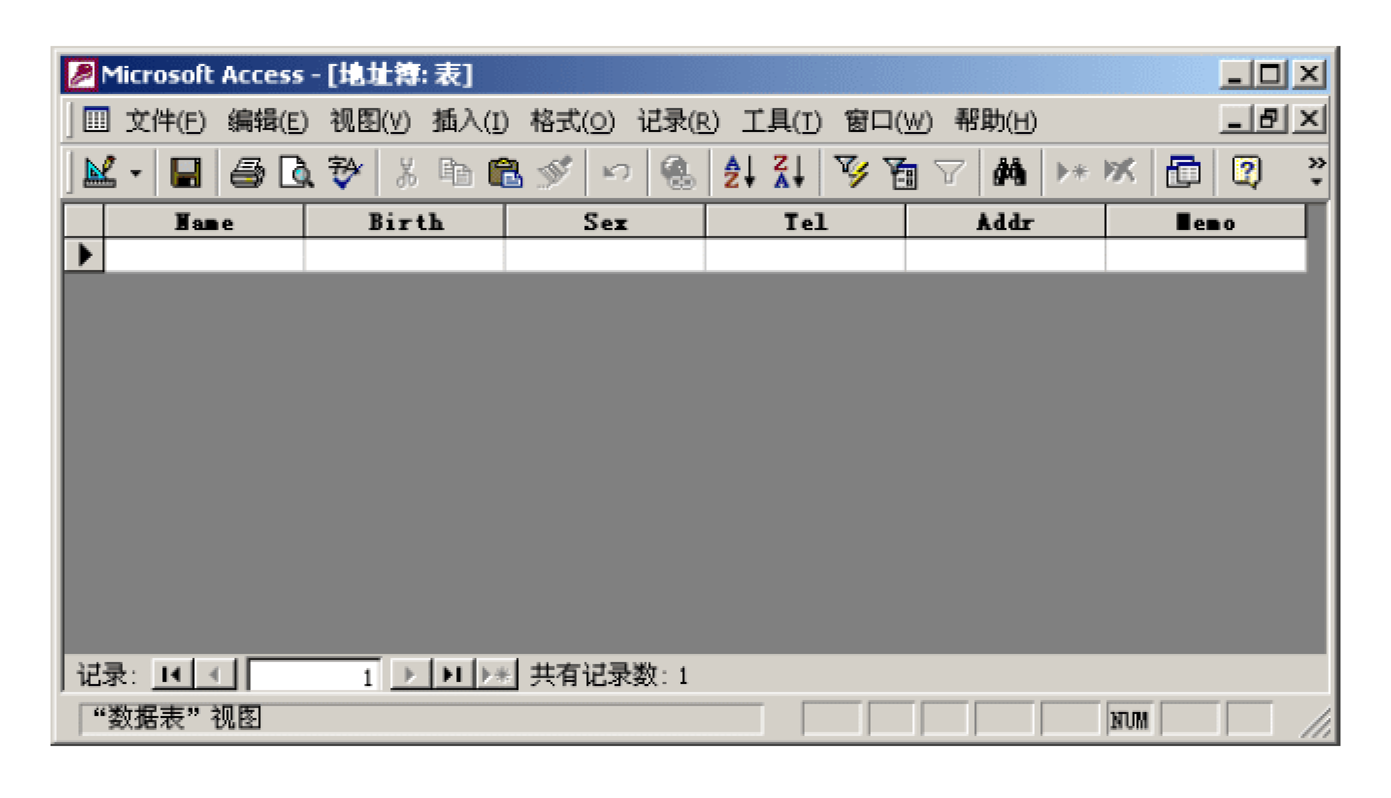


图 4.16　数据表的浏览窗口

在数据表的浏览窗口中，表头显示的就是定义的字段。在表头的下方是一空行，且每一列均对应一个单元格，这个空的单元格就是输入数据的地方。直接在各个单元格中输入相应的数据，如图 4.17 所示。

图 4.17　输入数据

一般来说，在添加数据的同时，需要注意以下三点。

(1) 输入的数据应与该字段所定义的数据类型一致，如在 Birth 字段对应的单元格中必须输入日期格式的数据。

(2) 必须保证主键不冲突，即对于定义为主键的字段，其数据不能重复。在本例中，数据表的主键为 Name 字段和 Birth 字段。在图 4.17 中，已经输入了 Name 字段为“张三”、Birth 字段为“1978-08-11”的数据，那么在后面的记录添加中，不能再有 Name 字段为“张三”、Birth 字段为“1978-08-11”的数据，否则系统将提示系统主键冲突，如图 4.18 所示。

图 4.18　系统提示主键冲突

(3) 对于输入的数据，不能违反其相应字段所定义的规则。例如，Name 字段所定义的大小为 20 个字符，也就是说 Name 字段对应的数据不能超过 20 个汉字。

4.3.2　数据表中记录的删除

对于数据表中记录的删除，可分为两种情况：删除单条记录和同时删除多条记录，其操作分别如下。

1. 删除单条记录

(1) 通过单击要删除的记录左边的记录选定器选择该记录，然后右击，在弹出的快捷菜单中选择【删除记录】命令，此时系统将弹出删除确认对话框，如图 4.19 所示。单击【是】按钮，即可删除该条记录。

(2) 选择要删除的记录，直接按 Delete 键，在弹出的删除确认对话框中，单击【是】按钮。这是一种简单的方法。

2. 删除多条记录

首先，单击要删除的第一条记录左边的记录选定器选择该记录，然后拖动鼠标向下移动，直至选择删除的其他记录。右击，在弹出的快捷菜单中选择【删除记录】命令，或直接按 Delete 键，此时系统将弹出删除确认对话框，提示所要删除的记录数，如图 4.20 所示。单击【是】按钮，删除所选择的多条记录。

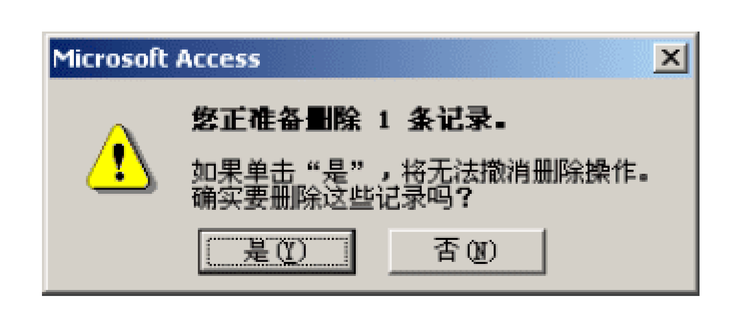


图 4.19　删除确认对话框(1)

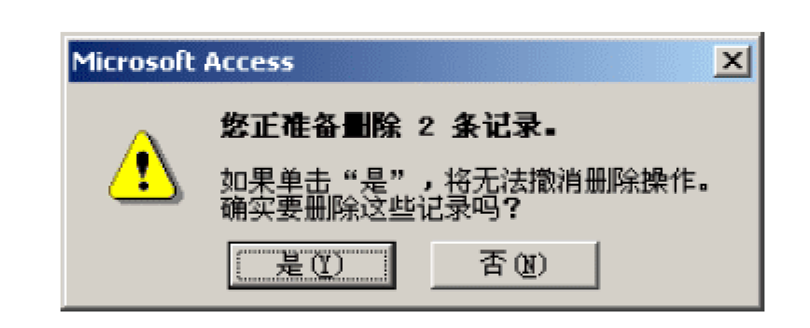


图 4.20　删除确认对话框(2)

一般来说，只能删除连续的多条记录，而无法同时删除非连续的多条记录。

4.3.3　数据表中记录的修改

在 Access 数据库中，修改数据表中的记录非常简单。由于数据表的浏览窗口本身就是一个全屏幕编辑器，只需将光标移到所要修改的数据处，即可修改数据。

值得注意的是，在 Access 数据库，修改记录的操作实质上是执行了以下两步操作。

(1) 删除原有记录。

(2) 将修改后的记录作为一条新记录添加至原记录的所在行。

因此，在修改记录时，同样需要注意添加记录时的三点事项(参见 4.3.1 节)。

4.4　数据库查询语言

4.3 节中介绍的对数据的操作都是在数据库的数据表浏览窗口中进行的。但在实际的编程应用中，很少直接在数据库中进行操作，更多的是通过数据库查询语言(SQL 语言)来执行对数据的操作。

SQL(Structured Query Language，结构化查询语言)是一种在关系数据库中的定义和操纵的语言。SQL 语言结构简洁、功能强大，且简单易学。目前，无论是 Oracle、Sybase、

SQL Server 等大型的数据库管理系统，还是 Microsoft Access、Visual Foxpro、PowerBuilder 等计算机上常用的数据库开发系统，都支持 SQL 语言。掌握好 SQL 语言，对于 ASP.NET 编程是非常重要的。

在实际编程应用中，可以通过 SQL 来执行以下任务。

- 查询数据表中的记录。
- 在数据表中添加记录。
- 在数据表中修改记录。
- 在数据表中删除记录。
- 创建、修改和删除数据表对象。
- 保证数据库的一致性和完整性。
- 控制对数据和数据对象的存取。

其中，前面 4 条是在编程应用中使用最多的。下面，分别介绍 SQL 语言在这 4 个方面的具体应用。

4.4.1　查询记录

在 SQL 语言中，SELECT 语句主要用来查询数据库中的记录。

1. SELECT 语句的基本语法

```
SELECT 列名 FROM 表名
WHERE 查询条件
ORDER BY 排序列 [ASC/DESC]
```

2. 注解

列名是指从 FROM 关键字后所指定的表中提取的列；列名可以为多个，用逗号分隔；列名也可使用“*”来代替，表示选择指定表中的所有列(按它们在表定义中的顺序进行排列)。

表名是指要从中获取记录的所在的表；表名可以为多个，用逗号分隔。

查询条件是用来对查询出来的记录进行过滤；如果表名仅有一个，则查询条件通常是对其中单个字段的值进行判断；如果表名有多个，则查询条件一般还会包含多个数据表的关联。

排序列是指对查询出来的记录结果进行排序的依据，其后使用关键字 ASC，表示升序；其后使用关键字 DESC，表示降序。

3. 使用技巧

在相应的列名后接 AS 子句可为该列指定别名，别名将显示在查询结果中，替代原有的列名。

在相应的表名后接 AS 子句可为表的名称指定别名。

在列名前可使用关键字 ALL/DISTINCT 来表示是选取重复记录的全部还是其中的一条，默认值为 ALL。

可在关键字 SELECT 和列名之间通过 TOP N 来限制返回记录结果的行数，其中 N 表

示所要返回的记录数。

在列名后使用 INTO+表名，可创建一个新表，并同时将查询结果插入到新表中；新表的字段将与查询结果中的字段一致。

在 WHERE 子句后可通过 GROUP BY +列名，来对查询结果按列名进行分组，并可使用 HAVING 子句对分组后利用聚合函数求得的值进行条件判断。

常用的聚合函数包括 COUNT(取记录数)、SUM(求和)、AVG(求平均值)、MAX(求最大值)、MIN(求最小值)。其中，SUM 和 AVG 仅针对数字列。

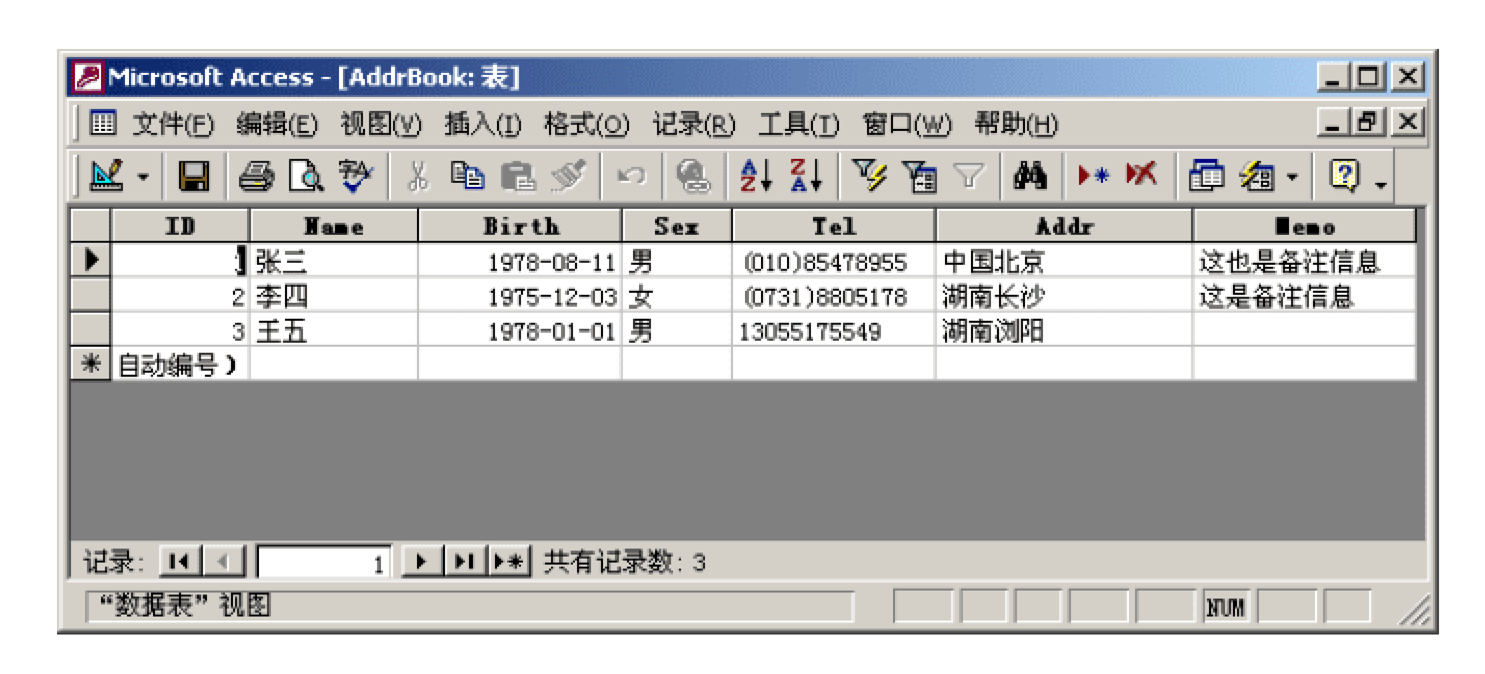
4. 相关实例

为了说明 SQL 语句的具体使用，创建了两个示例数据表：AddrBook 和 MyFriend，其中数据表 AddrBook 表示地址簿信息，其各字段的具体含义见表 4.2。

表 4.2　数据表 AddrBook 的字段含义

字 段 名	相关描述
ID	表示联系人的编号
Name	表示联系人姓名
Birth	表示联系人的出生日期
Sex	表示联系人的性别
Tel	表示联系人的联系电话
Addr	表示联系人的联系地址
Memo	表示联系人的备注信息

数据表 AddrBook 的数据如图 4.21 所示。



ID	Name	Birth	Sex	Tel	Addr	Memo
1	张三	1978-08-11	男	(010)85478955	中国北京	这也是备注信息
2	李四	1975-12-03	女	(0731)8805178	湖南长沙	这是备注信息
3	王五	1978-01-01	男	13055175549	湖南浏阳	
自动编号)						

图 4.21　数据表 AddrBook 中的数据

数据表 MyFriend 表示好友信息，其各字段含义见表 4.3。

表 4.3　数据表 MyFriend 的字段含义

字 段 名	相关描述
ID	表示好友的编号，该字段与数据表 AddrBook 中的 ID 字段相关联
RSDate	表示与好友认识的时间

数据表 MyFriend 中的数据如图 4.22 所示。

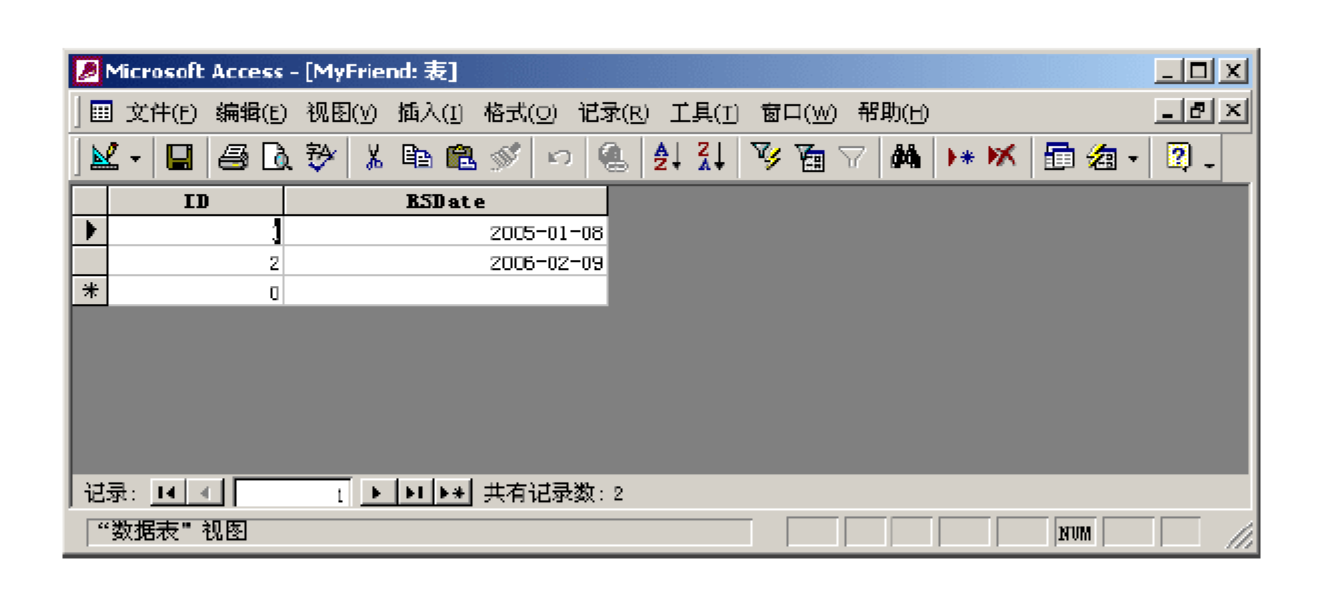


图 4.22　数据表 MyFriend 中的数据

示例数据有了，但如何在 Access 数据库中测试 SQL 语句的执行呢？可以通过 Access 数据库中的查询对象来实现，其具体操作如下。

首先，在数据库窗口左边的对象一栏中，选择【查询】命令，如图 4.23 所示。

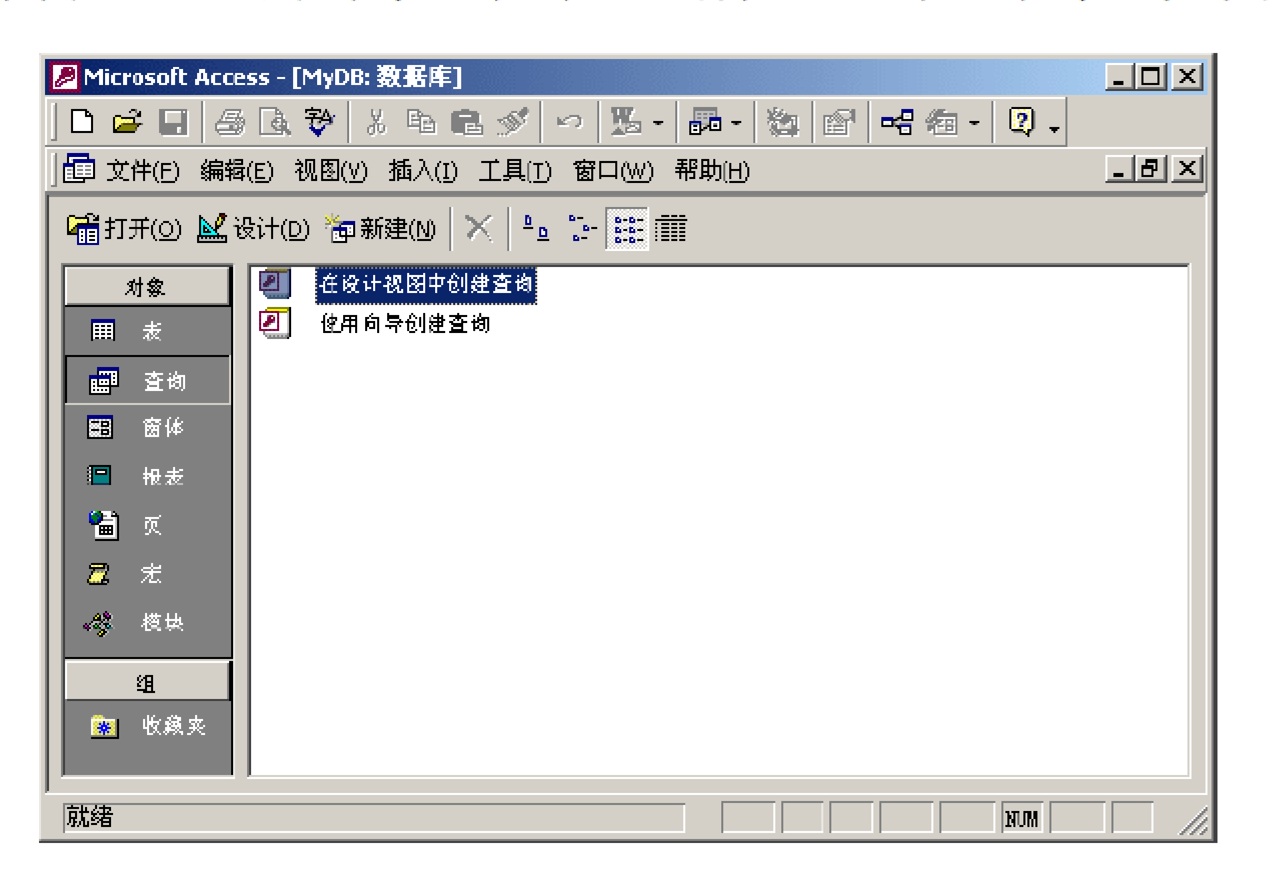


图 4.23　【查询】对象窗口

在【查询】对象窗口中，提供了【在设计视图中创建查询】和【使用向导创建查询】两种方式来创建查询，这里选择前者。双击【在设计视图中创建查询】，系统将打开查询窗口，如图 4.24 所示。

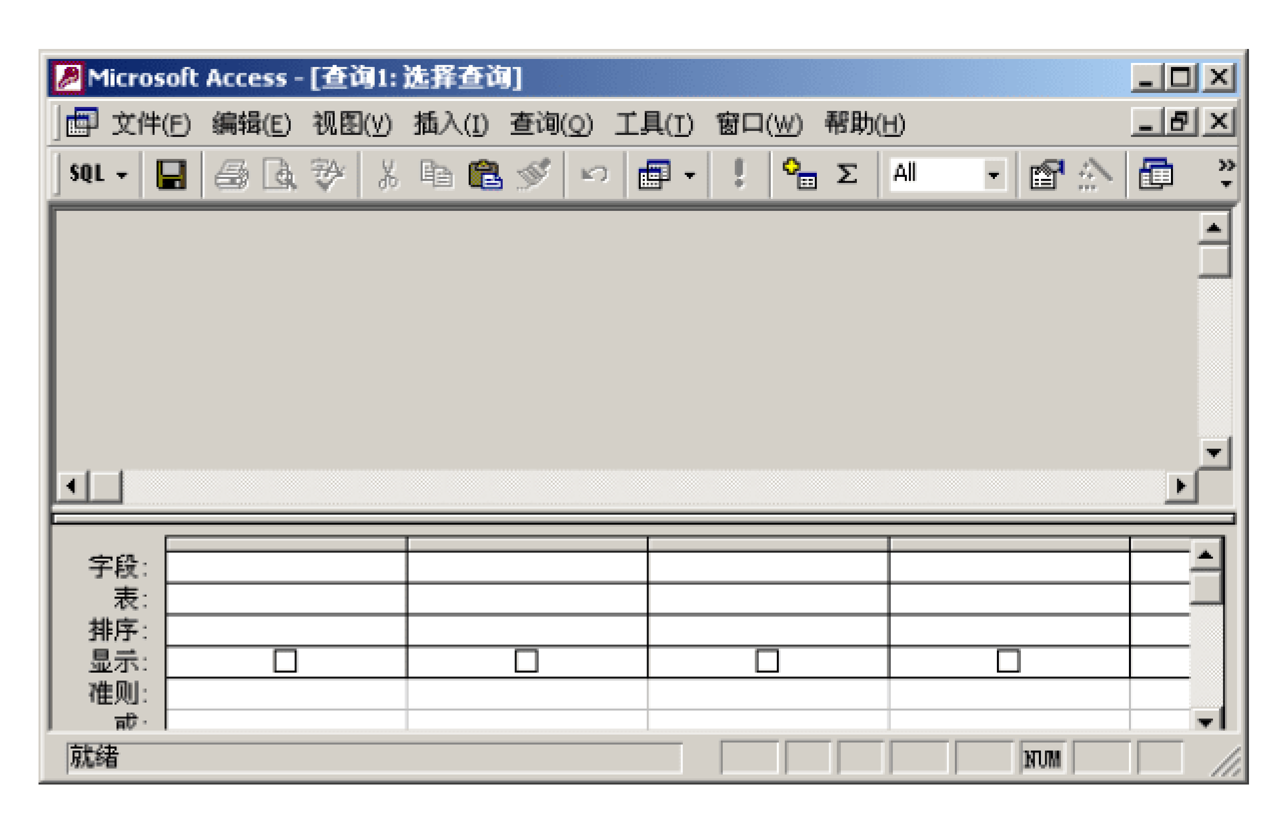


图 4.24　查询窗口

单击窗口工具栏最左侧的 SQL 按钮，在打开的下拉菜单中，选择【SQL 视图】命令，即可显示 SQL 查询窗口，如图 4.25 所示。

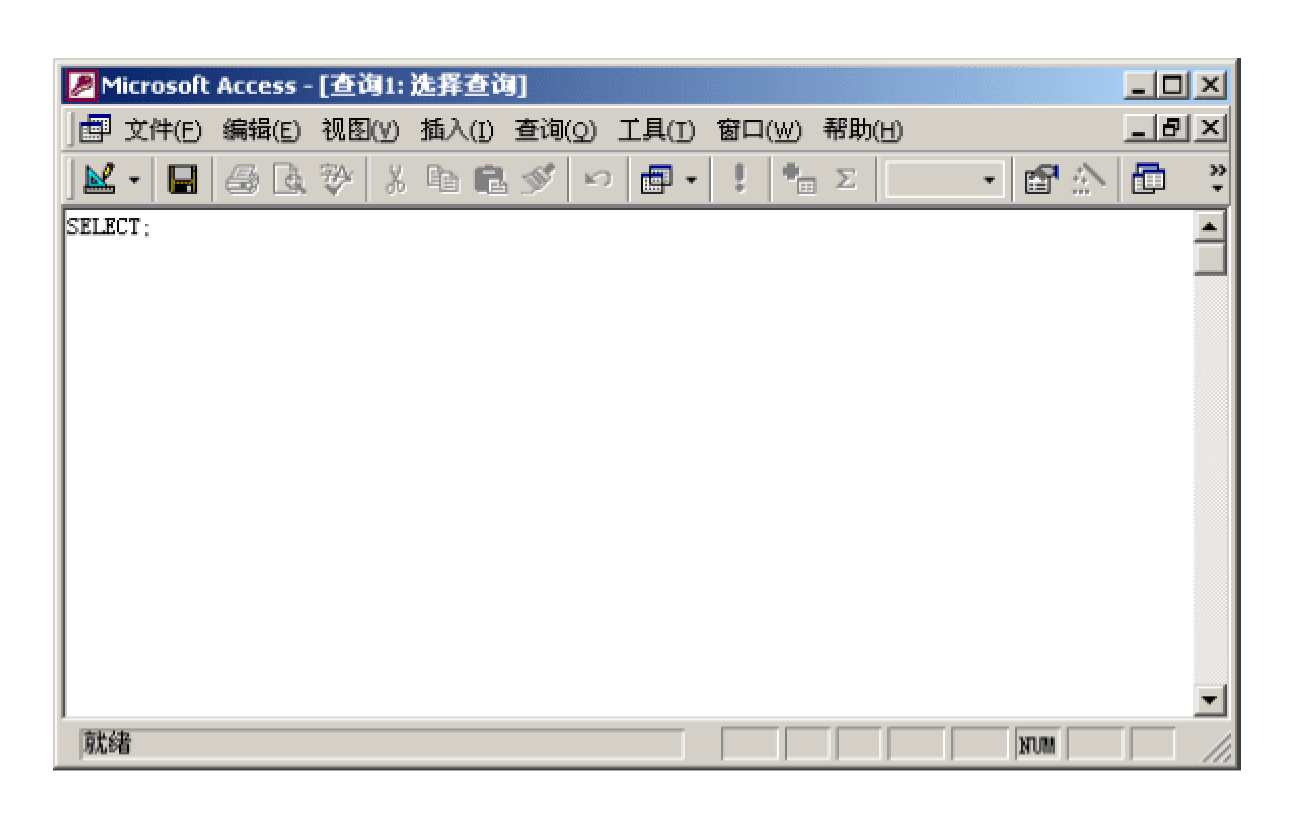


图 4.25　SQL 查询窗口

在 SQL 查询窗口下方的文本输入框中，直接输入 SQL 语句，单击工具栏中的【执行】按钮！，即可查看查询结果。对于后面所要提到的用于添加、删除和修改记录的 SQL 语句，均可使用此方法进行测试，对于测试结果则可打开相应数据表的浏览窗口进行查看。

接下来，介绍一些典型的 SELECT 语句示例。

例：查询所有的联系人信息，并按姓名进行排序。

```
Select * From AddrBook Order By Name
```

查询结果如图 4.26 所示。

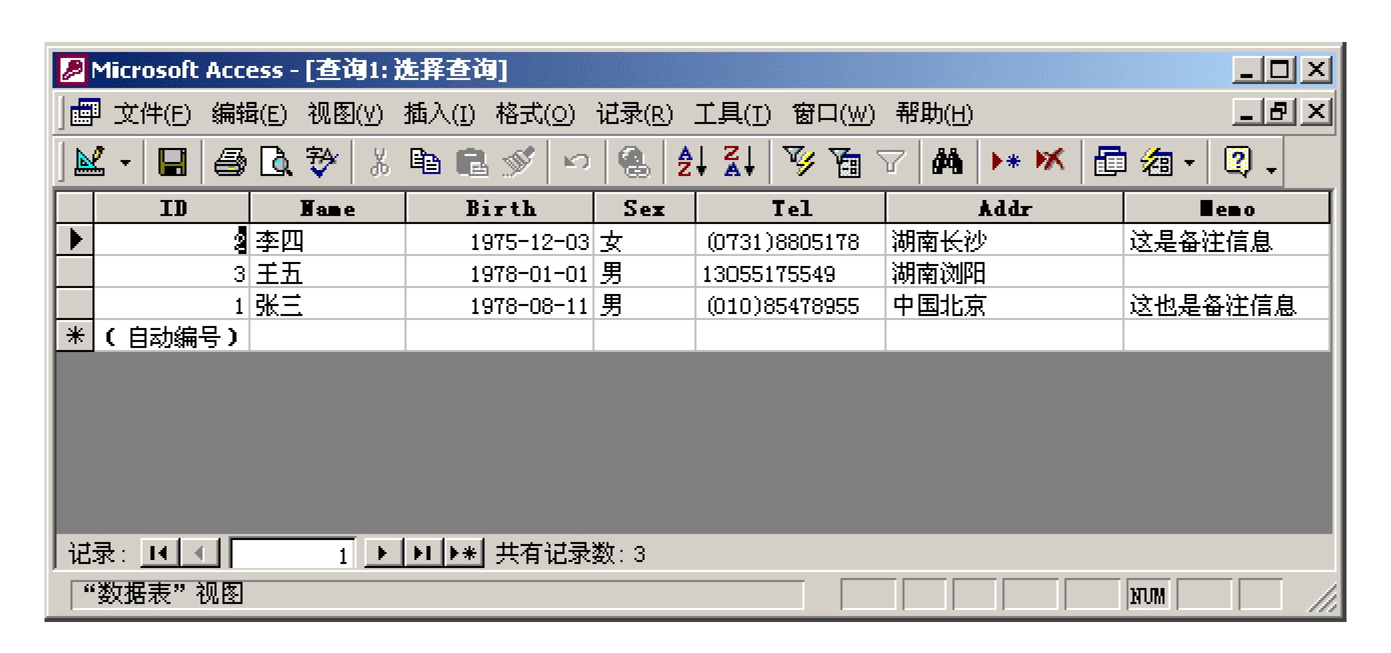


图 4.26　查询结果(1)

例：查询 1976 年以后出生的所有联系人信息，并按出生日期进行排序。

```
Select * From AddrBook where birth >=#1976-1-1# Order By birth
```

查询结果如图 4.27 所示。

图 4.27　查询结果(2)

例：查询好友的姓名、出生日期、联系电话、联系地址及认识时间，并按出生日期降序排序。

```
Select name,birth,tel,addr,rsdate
From AddrBook,MyFriend
where AddrBook.id=MyFriend.id order by birth desc
```

查询结果如图 4.28 所示。

图 4.28　查询结果(3)

例：查询不属于好友的联系人信息。

```
Select * From AddrBook Where id not in (Select id From MyFriend) Order By
Name
```

查询结果如图 4.29 所示。

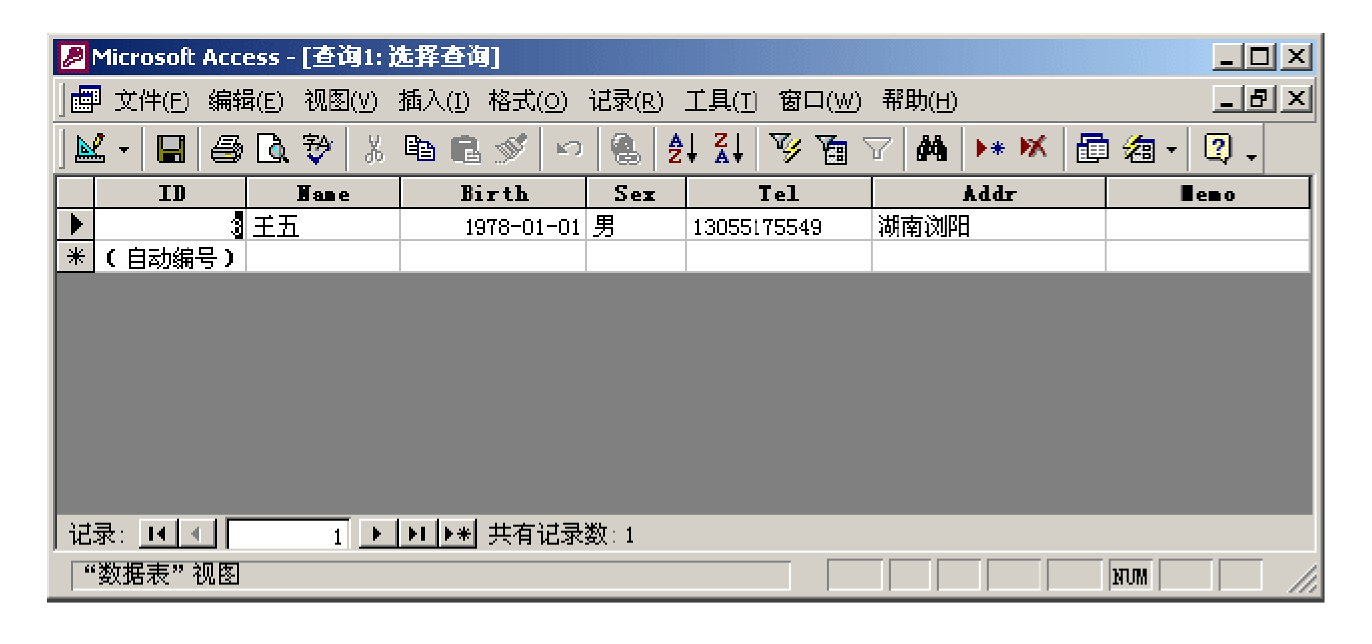


图 4.29　查询结果(4)

提示：请特别注意在此 SQL 语句中 Select 子句的应用。

4.4.2　添加记录

在 SQL 语言中，INSERT 语句主要用来向数据库中添加记录。

1. INSERT 语句的基本语法

```
INSERT INTO 表名 [(列名)]
VALUE (列值)
```

2. 注解

表名是指要插入记录的表；表名仅能为一个，不可以为多个。

列名表示要插入记录的列，可为多个列，用逗号分隔；列名可以忽略，此时表示所有插入表的所有列均需插入数据，除非该列允许空(NULL)值或有指定的默认值。

列值表示所要插入的数据；如果显示地指定了列名，则列值必须与其一一对应，其顺序也应与列名的顺序一致；如果省略了列名，则列值的顺序应与插入的表所定义的结构中的列的顺序一致。

3. 使用技巧

使用关键字 VALUE，一次性只能向数据表中插入一条记录；但是，用户可以使用 SELECT 子句来替代 VALUE (列值)，这样可以将 SELECT 子句所返回的记录一次性全部插入到指定的数据表中。

当只为表中的几个列指定值时，对没有指定值的列将存在以下 4 种情况。

(1) 如果该列设置了默认值，则将插入默认值；
(2) 如果该列的数据类型为自动编号，则将插入系统自动生成的一个数值；
(3) 如果该列的【必填字段】属性设置为【否】，则将插入 NULL 值；
(4) 如果该列的【必填字段】属性设置为【是】，且不存在默认值，则系统将提示错误信息，无法插入该行。

4. 相关实例

例：向 AddrBook 数据表中新增一条记录。

```
INSERT INTO AddrBook (Name,Birth,Sex,Tel,Addr,Memo)
VALUES ("陈剑", #1980-12-3#, "男","13607491111","湖南省长沙市高新开发区" ,"这是新加的记录")
```

提示：在插入列值时，对于文本型数据，其列值需要使用单引号括起来；对于数字型数据，其列值不需要单引号；而对于日期/时间型，其列值需要使用符号“#”括起来。

例：将不属于好友的联系人全部变为好友，即全部添加至 MyFriend 数据表中，其认识日期设置为 2006 年 1 月 1 日。

```
INSERT INTO MyFriend (ID,RSDATE)
SELECT ID,#2006-1-1# FROM AddrBook WHERE ID NOT IN (SELECT ID FROM MyFriend)
```

4.4.3　修改记录

在 SQL 语言中，UPDATE 语句主要用来修改数据库中的记录。

1. UPDATE 语句的基本语法

```
UPDATE 表名
SET 列名=表达式
WHERE 查询条件
```

2. 注解

表名是指包含了要更新的列的数据表。

“列名=表达式”表示使用表达式的值来对列名所指定的列进行更新；可以存在多个“列名=表达式”，同时对多个列进行更新，用逗号分隔。

查询条件用于过滤所要更新的记录；如果没有查询条件，则表中的所有记录均将被更新。

3. 使用技巧

可以使用多个表名，来指定从一个以上的表中进行查询，从而指定要更新的记录。

4. 相关实例

例：将 AddrBook 数据表中的姓名为“张三”记录的备注字段清空，即设置为空字符串。

```
UPDATE AddrBook SET Memo='' WHERE Name='张三'
```

例：将认识日期在 2006 年 1 月 1 日以后的所有联系人的备注信息更新为“新朋友”。

```
UPDATE AddrBook,MyFriend
SET Memo='新朋友'
WHERE rsdate>=#2006-1-1# AND AddrBook.id=MyFriend.id
```

4.4.4 删除记录

在 SQL 语言中，DELETE 语句主要用来删除数据库中指定的记录。

1. DELETE 语句的基本语法

```
DELETE FROM 表名
WHERE 查询条件
```

2. 注解

表名是指包含了要删除的记录的数据表；可以指定多个表名，用逗号分隔。

查询条件用于过滤所要删除的记录；如果没有查询条件，则表中的所有记录均将被删除。

3. 使用技巧

如果在 FROM 关键字后指定了多个表名，则必须在 DELETE 关键字后指定需要删除记录的是其中哪个表。

4. 相关实例

例：将 AddrBook 数据表中姓名为“张三”的记录删除。

```
DELETE FROM AddrBook WHERE Name='张三'
```

例：删除认识日期为 2006 年 1 月 1 日的所有联系人信息。

```
DELETE AddrBook.* FROM AddrBook INNER JOIN MyFriend ON
AddrBook.id=MyFriend.id WHERE Rsdate=#2006-01-01#
```

4.5 习　　题

(1) 练习 Access 2000 数据库及数据表的创建方法。

(2) 练习基本的 SQL 语句的语法及使用，包括 SELECT 语句、INSERT 语句、UPDATE 语句及 DELETE 语句等。

第 5 章　VB.NET 语言基础及 ASP.NET 网页基本结构

在前面 4 章中，介绍了 Dreamweaver 8 和数据库的相关基础知识。本章主要介绍 VB.NET 的语言基础以及 ASP.NET 的网页基本结构。

5.1　VB.NET 语言基础

在创建 ASP.NET 页面时，可以使用.NET 兼容的脚本语言来编写服务器端代码，如 VB.NET、C#等。VB 语言和 VBScript 脚本语言的普及，使得 VB.NET 也被众多 ASP.NET 程序员作为服务器端语言。

5.1.1　VB.NET 概述

VB.NET 是 Microsoft 推出的.NET 框架中的一个重要组成部分，它也是 Visual Basic(简写为 VB)的最新版本。与 VB 相比，VB.NET 在数据类型、基本语法、程序结构等方面均有很多相似之处。但作为.NET 框架的重要成员，VB.NET 在底层架构上有着明显的改进，如继承、重载、封装、接口、结构化的异常处理等。特别是对继承的实现，使得在 VB 6.0 中部分的对面向对象的支持转变为彻底的面向对象，也使其成为一种真正的面向对象的开发语言。

继承是面向对象系统中的一个重要概念，一门语言是否具有继承性常常被作为判断面向对象语言的关键标准。实际上，继承所表达的是一种类的相交关系，它使得某类可以继承另一类的特征和能力。从对象的角度上来说，如果一个对象能够获得另外一个对象的接口和方法，并且可以扩展这些接口和方法，就称这个对象继承了另一个对象。对于 VB.NET 来说，在继承关系中，一种类(派生类)从另一种类(基类)派生，派生类的声明空间隐式地包含了基类的所有可访问的非构造函数类型成员。VB.NET 中的所有类在默认情况下都是可以继承的。

在 VB.NET 中，实现了对对象的封装。从字面上来说，所谓封装是指将某事物包围起来，使外界不知道里面的实际内容，而仅仅知道其外在的表现形式。在程序设计中，封装的意思就是将一系列的函数和过程进行包装，然后提供相应的接口供其他程序进行调用。而对于接口里面各函数和过程的代码，外界程序不需要了解，也无法了解。这样，既提高了程序的方便、灵活调用，又保证了程序的安全性。

重载也是 VB.NET 中的一个重要功能。重载的含义是指通过为函数或运算符创建附加定义而使它们的名字可以重载。也就是说，相同名字的函数或运算符在不同的场合可以表

现出不同的行为。这也就意味着，使用重载特性，可以允许具有不同的数据类型的方法、属性或过程使用相同的名称，而执行不同的程序，从而为程序的编写和调用带来极大的方便，而这在 Visual Basic 6 中是根本不可能实现的。

此外，在异常处理方面，VB.NET 采用了结构化的异常处理。在 Visual Basic 6 中，通常采用“ON ERROR GOTO”语句来处理错误的出现。但在 VB.NET 中，对错误的处理采用 Try…Catch…Finally…End Try 语句来实现。该语句可以针对所出现的错误进行具体的判断，并执行相应的操作。这种结构化的处理，可以很好地控制错误的出现，而不会轻易导致程序崩溃。

总之，相对于 Visual Basic 6 来说，VB.NET 是一种更为成熟且功能更为强大的编程语言，相信它的使用会给用户在 ASP.NET 的编程中带来无限的乐趣。

5.1.2 VB.NET 数据类型

作为一种面向对象的编程语言，VB.NET 中的数据类型可分为值类型和引用类型两种。其中，值类型包括基元类型、枚举类型和结构类型等；引用类型包括类、字符串、标准模块、接口、数组和委托等。这里，主要介绍值类型中最常用的基元类型。

VB.NET 中的基元类型可分为整数值类型、浮点值类型、Boolean 值类型、Date 值类型、Char 值类型、String 数据类型等几类，下面分别详细叙述。

1. 整数值类型

整数值类型通常用来表示整数，它又包括 Byte、Short、Integer 和 Long 等几种类型，见表 5.1。

表 5.1 整数值类型

数据类型	映射结构	取值范围	占用空间	相关说明
Byte	System.Byte	0～255	1 个字节	无符号整数
Short	System.Int16	-32 768～32 767	2 个字节	有符号整数
Integer	System.Int32	-2 147 483 648～2 147 483 647	4 个字节	有符号整数
Long	System.Int64	-9 223 372 036 854 775 808 ～ 9 223 372 036 854 775 807	8 个字节	有符号整数

在使用整数类型时，必须注意最大与最小取值范围。在以此类型定义变量时，应确定该变量的取值不会超出相应整数类型的取值范围，否则极易导致溢出错误。

Byte 数据类型可以直接被转换为 Short、Integer、Long、Single、Double 或 Decimal 等数据类型。

Short 数据类型可以直接被转换为 Integer、Long、Single、Double 或 Decimal 等数据类型；可以通过在文本后直接追加字符 S 来将其强制转换成 Short 数据类型。

Integer 数据类型可以直接被转换为 Long、Single、Double 或 Decimal 等数据类型；可以通过在文本后追加字符 I 来将其强制转换成 Integer 数据类型；此外，在任何标识符后追加字符%都可将其强制转换成 Integer 数据类型。

Long 数据类型可以直接被转换为 Single、Double 或 Decimal 等数据类型；可以通过在文本后追加字符 L 来将其强制转换成 Long 数据类型；此外，在任何标识符后追加字符“&”均可将其强制转换成 Long 数据类型。

整数类型的默认值均为 0。

2. 浮点值类型

浮点值类型通常用来表示小数，它包括 Single、Double 和 Decimal 三种类型，见表 5.2。

表 5.2　浮点值类型

数据类型	映射结构	取值范围	占用空间
Single	System.Single	负值取值范围为： −3.4028 23 5E+38～−1.401 298E−45 正值取值范围为： 1.401 298E−45～3.402 823 5E+38	4 个字节
Double	System.Double	负值取值范围为： −1.797 693 134 862 315 70E+308～ −4.940 656 458 412 465 44E−324 正值取值范围为 4.940 656 458 412 465 44E−324～ 1.797 693 134 862 315 70E+308	8 个字节
Decimal	System.Decimal	不带小数点的数的取值范围为 0～+/−79 228 162 514 264 337 593 543 950 335 带 28 位小数的数的取值范围为： 0～+/−7.922 816 251 426 433 759 354 395 033 5 最小非零数为： +/−0.000 000 000 000 000 000 000 000 (+/−1E−28)	16 个字节

Single 数据类型可以直接被转换为 Double 或 Decimal 数据类型；通过在文本后追加字符 F 可将其强制转换成 Single 数据类型；此外，在任何标识符后追加字符“!”都可将其强制转换成 Single 数据类型。

Double 数据类型可以直接被转换为 Decimal 数据类型；通过在文本后追加字符 R 可将其强制转换成 Double 数据类型；此外，在任何标识符后追加字符“#”都可将其强制转换成 Double 数据类型。

与 Single 和 Double 两种数据类型不同的是，Decimal 是定点型，它是用来替代原 VB 6 中的 Currency 数据类型的；在文本后追加字符 D 可将其强制转换为 Decimal 数据类型；此外，在任何标识符后追加字符“@”都可将其强制转换为 Decimal 数据类型。

浮点类型的默认值与文本“0”等效。

3. Boolean 值类型

Boolean 值类型通常用于表示关系运算或逻辑运算的结果，其取值为 True 或 False。Boolean 值类型占 2 个字节，其对应的映射结构为 System.Boolean。

在将数据值数据类型转换为 Boolean 类型时，0 将被转换为 False，而其他非零值将会转换为 1。但是，将 Boolean 值转换为数值类型时，False 将被转换为 0，而 True 将被转换为-1。

Boolean 类型的默认值等于 False。

4. Date 值类型

Date 值类型通常用于表示日期或时间，其取值范围为 1 年 1 月 1 日到 9999 年 12 月 31 日的日期以及从凌晨 0:00:00 到晚上 11:59:59 的时间。

Date 值类型占 8 个字节，其对应的映射结构为 System.DateTime。

一般来说，Date 类型的数据必须写在两个“#”之间，如“# January 1,2006#”，而日期和时间的显示将取决于计算机的相关区域设置(此操作可在【控制面板】|【区域选项】中的【日期】和【时间]两个选项卡中进行)。

Date 类型的默认值等于#01/01/0001 12:00:00AM#。

5. Char 值类型

Char 值类型通常用于表示单个的 Unicode 字符，其取值范围为 0～65 535，其中每个数字均代表一个 Unicode 字符。

Char 值类型占 2 个字节，其对应的映射结构为 System.Char。

不能直接在 Char 数据类型和数值类型之间进行转换，但可以使用 AscW 和 ChrW 函数来执行转换操作。

通过在单字符字符串文本之后附加字符 C，可将其强制转换为 Char 数据类型。

Char 类型的默认值等于 ChrW(0)。

6. String 数据类型

String 数据类型通常用来表示字符串数据，它由一系列的 Unicode 字符组成，一个字符串中可以存储将近 20 亿(2^{31})个 Unicode 字符。

String 数据类型占 16 个字节，其对应的映射结构为 System.String。

通过在任何标识符后追加字符“$”可将其强制转换成 String 数据类型。

String 数据类型的默认值为空引用。

5.1.3　VB.NET 变量与常量

变量，是指用来存储在程序运行过程中所出现的一些临时数据的内存单元。在程序中只需通过语句操纵变量，即可操纵该变量所对应的内存单元中的数据。

在 VB.NET 中，依据声明的级别和范围，变量可分为局部变量、模块变量、共享变量、实例变量、块级变量等。

局部变量是指在过程内部所声明的变量，其有效范围仅在声明它的过程中。

模块变量是指在模块级声明的，位于模块内部但不在该模块的任何过程内的变量，其有效范围是在其所在模块的整个命名空间。

共享变量和实例变量均是在类或结构中所定义的非局部变量，其有效范围在声明它的类或结构中。如果通过关键字 Shared 来声明，则该变量为共享变量，否则为实例变量。对

于共享变量，它存在于由类或结构的所有实例所共享的一个副本中；而对于实例变量，必须为类或结构的每个实例单独创建该变量的副本。

块级变量是指在一个块内部所声明的变量，其有效范围仅在其所在的块的内部有效。所谓块，是指以 End、Else、Loop 或 Next 语句终止的语句集合，如 For...Next 或 If...Then...Else...End If 结构。

变量在使用之前，必须对其进行声明并赋值。变量的声明使用 Dim 语句，如 Dim I As Integer，该语句是声明一个整型变量。在 VB 6 中，如果在源程序的最前面添加一句 Option Explicit on，则表示在该文件中的所有变量均需要声明后方可使用。而在 VB.NET 中，默认为使用 Option Explicit，也就是说对变量必须进行强制声明。事实上，对变量进行声明是一个良好的编程习惯，有助于减少程序错误的发生。

对于每个变量，均有两个属性：变量名和数据类型。前者，是用来在程序中引用该变量的标识；后者，则代表了该变量可存储的数据类型。变量名由用户自行定义，但必须符合一定的规则，其规则如下。

- 名称不能是 VB.NET 中的关键字，如 Double、Char、Sub 等；
- 名称必须以字母或下划线开头，如 strName，_myVar 等；
- 名称不能包含标点符号或特殊字符，如空格、斜杠、运算符等；
- 在同一个作用域内，名称必须是惟一的；
- 名称不能超过 255 个字符。

在为变量进行命名时，养成良好的命名习惯是非常重要的，这将有助于代码更加易于阅读和维护。一个好的命名风格，应遵循以下规则：数据类型+标识名称。其中，数据类型是指像 Integer、String 等数据类型的小写的三码简写，见表 5.3。而标识名称是变量的主体，建议使用以英文大小写搭配，容易记忆且能代表变量用意的名称。

表 5.3　不同数据类型的命名规则

数据类型	三码简写	变量命名示例
Byte	Byt	bytType
Boolean	Bln	blnIsTrue
Integer	Int	intAge
Long	Lng	lngNumber
Single	Sng	sngPrice
Double	Dbl	dblIncome
Short	Shr	shrYear
Decimal	Dec	decSalart
Date	Dat	datBirth
Object	Obj	objCmd
String	str	strName

变量的赋值有两种方式，一种是在定义的同时对其进行赋值，如：

```
Dim intMyAge As Integer=31 '声明整型变量 intMyAge，并对其赋初值 31
```

另一种是先定义变量，在随后的程序中再对其进行赋值，如：

```
Dim intMyAge As Integer '声明整型变量 intMyAge
intMyAge=100  '对变量 intMyAge 进行赋值
```

此外，对于变量来说，存在一个生存周期，即变量可供使用的那段时间。对于使用关键字 Dim 所声明的局部变量，只在其过程正在执行期间存在；当过程终止时，其所有的局部变量均会消失，值也会丢失。但对于使用关键字 Static 所声明的局部变量，即使过程结束，其变量依然存在且保留其值。

所谓常量，是用来存储在整个应用程序的执行过程中均保持不变的值，其命名规则、类型、作用范围均与变量相同。使用 Const 关键字可以定义常量，常量必须在定义时赋予初值，且在运行期间不能修改，如：

```
Dim Const strMyName As String="Hyh"
```

这里定义了一个字符串常量 strMyName，其值为 Hyh，该值在程序运行期间无法修改。

5.1.4　VB.NET 的内置函数

在 VB.NET 中提供了大量的函数，这些函数有助于我们在程序中快捷实现一些简易的功能。依据函数的功能，VB.NET 中的内置函数可分为字符串函数、数学函数、日期时间函数、类型转换函数等。

1. 字符串函数

字符串函数主要包括针对字符串进行处理的函数，见表 5.4。

表 5.4　字符串函数

函 数 名	说　明	示　例
Asc	获取字符串的第一个字符的 ASCII 码	Asc("A")，返回 65
Chr	获取与指定的 ASCII 码相对应的字符	Chr(65)，返回"A"
Len	获取字符串的长度	Len("ABCD")，返回 4
Trim	删除字符串中的所有空格，包括起始和末尾空格	Trim(" name ")，返回"name"
Ltrim	删除字符串起始的空格	LTrim(" name ")，返回"name "
Rtrim	删除字符串末尾的空格	RTrim(" name ")，返回" name"

续表

函 数 名	说　明	示　例
Filter	搜寻字符串数组中的指定字符串，凡是数组元素中含有指定字符串，会将它们结合成新的字符串数组并传回	Dim strName(2) as String strName(0)="胡金" strName(1)="张三" strName(2)="胡萝卜" Dim retStr() as String retStr=Filter(strName,"胡", True, CompareMethod.Text) 返回数组：["胡金","胡萝卜"]
Format	获取指定字符串中所包含的指令设置格式字符串	Format(0.5,"0.00%")， 返回 0.50%
FormatCurrency	获取格式为金额型态的表达式，该金额型态使用系统控制面板中所定义的货币符号	FormatCurrency(1234.56, , , TriState.True, TriState.True) 返回$1,234.56
FormatDateTime	获取格式化的日期或时间数据	Dim DDate As DateTime DDate= #3/12/1999# retStr= FormatDateTime(DDate, dateFormat.LongDate) 返回 Friday, March 12, 1999
FormatNumber	获取格式化的数值数据	FormatNumber(1234, 2, , , TriState.True) 返回 1,234.00
FormatPercent	获取转换为百分比格式的数值数据	FormatPercent(0.23) 返回 0.23%
InStr	获取一个字符串在另一个字符串中的第一个匹配项的位置	InStr(1,"abcd","c")，返回 3
Join	通过指定的分隔符，将字符串数组中的项合并成一个字符串	Dim exStr(2) as String exStr(0)="aa" exExStr(1)="bb" exStr(2)="cc" RetStr=Join(exStr,",") 返回 aa,bb,cc
Lcase	将字符串转换为小写字体	Lcase("aBcD")，返回 abcd
Ucase	将字符串转换为大写	Ucase("aBcD")，返回 ABCD
Left	获取字符串从左边开始指定长度的字符串	Left("abcdef",3)，返回 abc
Right	获取字符串从右边开始指定长度的字符串	Right("abcdef",3")，返回 def

续表

函 数 名	说　明	示　例
Mid	从一个字符串中返回包含从指定位置开始、指定数量字符的字符串	Mid("abcdef",3,4)，返回 cdef
Replace	返回一个字符串，其中的指定子字符串已被另一个子字符串所替换	Replace("abcd","b","c") 返回 accd
Space	取得指定数量的空白字符串	StrName="Hu" & Space(2) & "YongHui" 返回"Hu　YongHui"
Split	通过指定的条件字符串将字符串分割为字符串数组	Dim exStr as String="a,b,c" Dim getStr() as String GetStr=Split(exStr,",") 返回数组：["a","b","c"]
StrReverse	获取与指定字符串中顺序相反的字符串	StrReverse("abcd")，返回 dcba
Val	将代表数字的字符串转换为数值型态，当遇到第一个无法解释为数字、数字修饰符、数字标点或空格的字符时将停止转换	Val("23 45 and 67")，返回 2345

2. 数学函数

在.NET 框架的 System.Math 类中，提供了用于数学计算的相关函数，包括三角函数、对数函数及其他常用数学函数，见表 5.5。

表 5.5　数学函数

函 数 名	说　明	示　例
Abs	获取数值的绝对值	Abs(−12)，返回 12
Sin	获取一个角度的正弦值	Dim dblRes as Double DblRes=Sin(30*(pi/180)) 返回 0.5
Cos	获取一个角度的余弦值	Cos(pi)，返回−1 pi 是 Math 类中的常量，表示 180°
Tan	取得某个角度的正切值	Dim dblRes as Double DblRes=tan(45*(pi/180)) 返回 1
Atan	获取包含角度的 Double 值，该角度的正切值是指定的数值。返回值为正，表示角度在 X 轴的逆时针方向上；返回值为负，表示顺时针角度。将该返回值乘以 180 再除以 pi，就可以将弧度转换为度	Dim dblPI as Double DblPI=4 * Atan(1) 返回值：3.141 592 653 589 79，这也就是 pi 的值

续表

函 数 名	说　明	示　例
Int	去掉指定数字的小数部分，返回整数部分	Int(7.56)，返回 7 Int(−8.2)，返回−9
Fix	去掉指定数字的小数部分，返回整数部分；Fix 与 Int 函数的不同在于，如果指定参数为负数，则 Int 函数返回小于或等于指定参数的第一个负整数，而 Fix 函数则返回大于或等于指定参数的第一个负整数	Fix(7.56)，返回 7 Fix(−8.2)，返回−8
Floor	返回小于或等于指定值的最大整数	Floor(−1.24)，返回−2 Floor(2.56)，返回 2
Log	取得数值的自然对数	Log(1)，返回 0
Exp	返回 e 的指定次幂，该函数为 Log 函数的反函数	Exp(0)，返回 1
Max	返回两个指定数字中较大的一个	Max(12,24)，返回 24
Min	返回两个指定数字中较小的一个	Min(12,24)，返回 12
Round	返回最接近指定值的数字	Round(3.56)，返回 4 Round(3.45,1)，返回 3.4 Round(3.46,1)，返回 3.5
Oct	将指定数值转换为八进制值	Hex(459)，返回 713
Hex	将指定数值转换为十六进制值	Hex(459)，返回 1CB
Sign	判断数值内容是正数还是负数，正数返回 1，负数返回−1，为零则返回 0	Sign(−123)，返回−1 Sign(123)，返回 1 Sign(0)，返回 0
Sqrt	获得指定数值的平方根	Sqrt(9)，返回 3
Str	将数字转为字符串后传回；当数值为正数时，将始终为其前导符号保留一个空格	Str(123)，返回“123” Str(−123)，返回“−123”

提示： 要使用数学函数，需在源代码的顶部添加以下代码，以便将 System.Math 命名空间添加至项目中。

```
<%@ Import NameSpace="System.Math" %>
```

3. 日期时间函数

在 VB.NET 中，提供了大量用于对日期和时间进行处理的函数，见表 5.6。

表 5.6　日期时间函数

函 数 名	说　明	示　例
Now	返回系统当前的日期和时间	Now()，返回值为 2006-9-25 17:32:35

续表

函 数 名	说　明	示　例
Today	返回系统当前的日期，不含时间	Today()，返回 2006-9-25
Year	返回指定日期的年度	Year("2006-9-25")，返回 2006
Month	返回指定日期的月份	Month("2006-9-25")，返回 9
MonthName	返回指定月份的月份名称，该名称不仅取决于输入参数，也取决于控制面板中指定的“区域选项”设置	MonthName(9)，返回“九月”
Day	返回指定日期的天	Day("2006-9-25")，返回 25
WeekDay	返回指定日期是一个星期的第几天，默认从星期天算起	WeekDay("2006-9-25") 返回 2，即星期一
WeekDayName	指定一个星期的第几天参数，返回其所代表的星期名称	WeekDayName(2) 返回“星期一”
Hour	返回指定时间的小时字段	Hour("2006-9-25 17:32:35") 返回 17
Minute	返回指定时间的分字段	Minute("2006-9-25 17:32:35") 返回 32
Second	返回指定时间的秒字段	Second("2006-9-25 17:32:35") 返回 35
DateAdd	返回在指定日期或时间上添加指定时间间隔的日期或时间值	DateAdd("d",10,"2006-9-25") 在指定日期上添加 10 天，其返回值为 2006-10-5
DateDiff	计算两个指定的日期或时间之间的差值	DateDiff("d","2006-7-2","2006-9-25") 计算两个日期之间相差的天数，返回值为 85
DatePart	获取指定的日期或时间中的某个部分的值，如年、月、日、时、分、秒、季度、星期、周等	DatePart("q","2006-9-25") 计算该日期所属的季度，返回值为 3，表示第三季度
DateSerial	将接收的年、月、日参数合并为日期格式的数据	DateSerial(2006,9,25)，返回值为 2006-9-25
TimeSerial	将接收的时、分、秒参数合并为时间格式的数据	TimeSerial(15,20,45)，返回值为 15:20:45
DateValue	用于将字符串转变为日期格式	DateValue("February 12,2006")，返回值为 2006-2-12
TimaValue	用于将字符串转变为时间格式	TimeValue("5:32:35 PM")，返回值为 17:32:35

4. 类型转换函数

在编程中，经常需要将一些数据由某种数据类型转变成另一种数据类型，这个过程也叫做类型转换。在 VB.NET 中，就提供了大量的类型转换函数，为用户的操作提供了便利，见表 5.7。

表 5.7　类型转换函数

函数名	说　明	示　例
CBool	将字符串表达式或字符串表达式转换为 Boolean 值；当表达式的值为非零时，返回 True；当表达式的值为零时，返回 False	Dim A,B As Integer A=10 B=10 CBool(A=B)，返回 True
CByte	将指定的数值转换为字节，其参数必须为 0～255 范围内的一个数值	CByte(98.58)，返回 99
CChar	将指定的表达式转化为字符类型，如果表达式为字符串，则仅转换其第一个字符；表达式的值不能为数字	CChar("ABCD")，返回 A
CDate	将指定的字符串表达式转换为日期类型	CDate("February 15,2006") 返回值为：2006-2-15
CDbl	将指定的数值表达式转换为 Double 类型	CDbl(0.78*26.12)，返回 Double 类型的数值 20.3736
CDec	将指定的数值表达式转换为 Decimal 类型	Dim dblVar as Double Dim decVar as Decimal dblVar=123 456.012 3 DecVar=Cdec(dblVar)，将 Double 类型的数值通过 CDec 函数赋值给 Decimal 类型变量
CInt	将指定的数值表达式转换为 Integer 类型	CInt(1 234.56)，返回 1234
CLng	将指定的数值表达式转换为 Long 类型	CLng(34 567.89)，返回 34568
CObj	将指定的数值表达式转换为 Object 类型	Dim dblVar as Double Dim objVar as Object dblVar=1.234 567 objVar=Cobj(dblVar)
CShort	将指定的数值表达式转换为 Short 类型	Dim bytVar as Byte Dim shrVar as Short bytVar=100 shrVar=Cshort(bytVar)
CSng	将指定的数值表达式转换为 Single 类型	CSng(1.232 156 7)，返回 1.232 16

续表

函 数 名	说　明	示　例
CStr	将指定的表达式转换为 String 类型；当表达式为 Boolean 值时，返回包含 True 或 Flase 的字符串；当表达式为日期类型时，返回以系统的短日期格式表示的 Date 值的字符串；当表达式为数值时，返回表示该数字的字符串	CStr(5=5)，返回 True CStr(123.45)，返回 123.45

5. 其他函数

除了以上类型的函数外，在 VB.NET 中还有一些经常需要使用的函数，见表 5.8。

表 5.8　其他函数

函 数 名	说　明	示　例
UBound	返回数组的指示维度的最大可用下标	Dim Array1(5) As Integer Dim Array2(7,10) As Integer Dim highest As Integer UBound(Array1)，返回 5 UBound(Array2,1)，返回 7 UBound(Array2,2)，返回 10
Iif	根据指定表达式的值，返回两个对象中的一个	Iif(5>4,"Large","Small") 返回 Large Iif(5<4, "Large","Small") 返回 Small
IsArray	返回一个 Boolean 值，指示变量是否为一个数组	Dim MyArray(4)　As Integer Dim MyStr As String IsArray(MyArray)，返回 True IsArray(MyStr)，返回　False
IsDate	判断表达式内容是否为日期/时间类型，如果是则返回 True，反之则为 False	IsDate("1234")，返回 False IsDate("2006-1-1")，返回 True
IsNumeric	判断表达式内容是否为数值类型，如果是则返回 True，反之则为 False	IsNumeric("sdkf")，返回 False IsNumeric("222")，返回 True

5.2　ASP.NET 网页基本结构

5.2.1　ASP.NET 中的代码块

代码块是 ASP.NET 页面中的一个重要部分，它包含了实现页面功能的关键代码。从某种程度上来说，可以将代码块分为代码声明块、代码呈现块以及相关的页面编译指令和命名空间等。

1. 代码声明块

在第 1 章的上机指导中，创建了一个简单的 ASP.NET 页面，同时在页面中添加了一段 Page_Load 事件代码，如图 5.1 所示。

```
<%@ Page Language="VB" ContentType="text/html"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<script runat="server">
Sub Button1_Click(Src As Object, E As EventArgs)
  LblInfo.text="您所输入的是：" & TextBox1.Text
End Sub
</script>
<title>我的第一个ASP.NET页面</title>
```

图 5.1　代码声明块

此段代码放置在标签<Script Runat="Server" >与标签</Script>之间。事实上，对于 ASP.NET 中的所有事件以及自定义的过程和函数等，其代码均需放置在<Script Runat="Server">和</Script>标签对之间，这些代码就是所谓的代码声明块。

代码声明块主要用于定义应用程序中的成员变量、事件处理程序或用户自定义的子程序和函数等。在 ASP.NET 的<Script>标签中，Runat 属性是不可缺少的，且其值必须为 Server，它表示代码块是在服务器上运行的。如果没有设置 Runat 属性，则 ASP.NET 将会把这段代码看成是客户端代码，并将其直接发送至浏览器进行显示，而不会去编译执行它。

从理论上来说，代码声明块可以放置在 ASP.NET 页面中的任何位置。但是，作为一个好的习惯，一般来说应将其与 HTML 代码尽可能地分开。这样，在对代码进行跟踪调试时将会更为方便，同时代码也显得更为清晰，易于维护。

2. 代码呈现块

来看下面一段代码：

```
<body>
  <center>当天日期：<%=now().toString()%></center>
</body>
```

这是一段 HTML 代码，它将在页面上显示系统的当前时间。其中，now()是 ASP.NET 中内置的时间函数，用于获取系统的当前时间。这段代码被放置在标签<%=与标签%>之间，通常被称为代码呈现块。

代码呈现块是在页面显示时执行的，它一般包括两种样式：内联代码和内联表达式。内联代码放置的是一段独立的用于控制程序执行顺序的代码，如 If 语句、While 语句等；而内联表达式放置的是一个表达式，通常以标签<%=和标签%>的形式显示，如上述代码。

代码呈现块与代码声明块较为类似，甚至可以将代码呈现块放置在代码声明块中，它们将以类似的方式运行。但两者之间还是存在一定的差别。首先，代码声明块将被编译为机器码(首先被编译为 MSIL)，这使得它比代码呈现块的速度更高，稳定性更好。其次，两者的处理次序不同，代码声明块是在被调用时方才执行，而代码呈现块是在页面加载显示时便已执行。此外，代码声明块是集中在一起的，而代码呈现块则比较分散。一般来说，

在使用两种代码形式均可实现页面指定功能的前提下，推荐使用代码声明块。

需要注意的是，在 ASP.NET 中的代码呈现块中，不能定义过程和函数。所有的过程和函数的定义都必须放置在代码声明块中，而不能放置在代码呈现块中，这一点与 ASP 是不同的。

3. 页面编译指令

页面编译指令是指当页编译器在处理 ASP.NET 页面时所使用的设置，即指示 ASP.NET 如何处理该页面。当使用页面编译指令时，一般是将其包含在文件的开头。对于每个页面编译指令，均可包含一个或多个特定于该指令的属性。

在图 5.1 所示的代码块中，第一行代码为<%@ Page Language="VB" ContentType="text/html"%>，这就是所谓的页面编译指令。

在 ASP.NET 中，常用的页面编译指令包括@ page 指令、@Import 指令、@Implements 指令、@Register 指令、@Assembly 指令、@OutputCache 指令以及@Reference 指令等。

其中，@page 指令是最常用的一个页面编译指令，它主要用于定义 ASP.NET 页分析器和编译器使用的页特定属性。该指令只能在 Web 窗体页中使用且每个页面文件只能包含一个@page 指令。

@Import 指令主要用于将指定的命名空间显式导入到 ASP.NET 页中，以便在页面中使用该命名空间所包含的所有类和接口。

@Implements 指令表示当前页或用户控件所指定的.NET 框架接口。

@Register 指令用于将别名与命名空间及类名关联起来，以便在自定义服务器控件语法中使用简明的表示法。

@Assembly 指令用于在编译过程中将程序集链接到当前页，以便程序集的所有类和接口都可用在该页中。

@OutputCache 指令表示以声明的方式控制 ASP.NET 页或页中包含的用户控件的输出缓存策略。

@Reference 指令是指以声明的方式指示另一个用户控件或页源文件应该被动态编译并链接到在其中声明该指令的页。

4. 命名空间

命名空间是微软在.NET 框架中新引入的一个专业术语，其作用是将多个类组成在逻辑上相关的一些单元，以便在.NET 中进行引用。每个命名空间均包含了可在程序中使用的类型，如类、结构、枚举、接口等。

命名空间的一个主要功能是对相关类型进行逻辑分组。例如，命名空间 System.Web 包含了所有管理 Web 请求的低级执行的 ASP.NET 类；命名空间 System.IO 包含了所有用来处理输入/输出操作的类。命名空间不对类型进行物理分组，这也是它与过去经常所使用的类，如 DLL 动态链接库、COM 对象或 OCX 模块等的主要区别。

减少名称冲突是命名空间的另一个主要功能。事实上，在面向对象的应用中，很多人可能会使用相同的类名称，而命名空间减少了这种名称冲突的可能性。因为在.NET 中，作为一个有效的类名称，是在命名空间的名称上再加上具体的类的名称。

对于命名空间的引用，是通过页面编译指令@Import 来完成的。

5.2.2　ASP.NET 中的表单

表单(Form)是 ASP.NET 开发中的一个重要组成部分，可以说没有表单就没有 ASP.NET Web 编程模型。表单主要用于实现对用户输入数据的提交。在 ASP 中，对表单数据的提交是将其提交至指定的页面中。而对于 ASP.NET，Form 可以提交至本身，同时 ASP.NET 模型提供了控件状态管理和 PostBack(回发)事件。

在 ASP.NET 的 Form 中，必须包含 Runat 属性，同时其值为 Server。如果在页面添加了一个 Form，而不含 Runat 属性，则该 Form 将会完全被 ASP.NET 忽略，并作为纯粹单一的 HTML 而直接呈现。

一般来说，在一个 ASP.NET 页面中，仅能包含一个服务器端的 Form 标记(即设置了 Runat 属性为 Server 的 Form)。如果有多个，则页面将会显示异常错误。但事实上，一个 ASP.NET 页面是可以包含任意数量的服务器端 Form，只是同一时间仅能有一个 Form 可见并呈现。这一点，是很多人所不太注意的。比如说，在页面上添加三个带有 Runat=Server 属性的 Form，但是仅有一个 Form 的 Visible 属性被设置为 True。此时，可在代码中通过设置其他 Form 的 Visible 属性，来在页面运行时动态改变当前所呈现的 Form。认识到这一点，将会对以后的编程有所帮助。

5.2.3　ASP.NET 中的服务器控件

在 ASP.NET 中，提供了大量的服务器控件供用户使用，掌握这些控件的使用是学好 ASP.NET 的一个重要前提。ASP.NET 中的服务器控件可分为 HTML 服务器控件和 Web 服务器控件。

HTML 服务器控件是在 HTML 标记的基础上衍生出来的控件，它允许通过程序直接控制 HTML 标记的属性。这些控件位于 System.Web.UI.HtmlControls 命名空间中，是从 HtmlControl 基类中直接或间接派生出来的。

HTML 服务器控件必须包含有“Runat="Server"”属性，以便 ASP.NET 能够将其视为服务器控件进行分析和处理。此外，HTML 服务器控件还必须驻留在具有“Runat="Server"”属性的<Form>标记中。

表 5.9 列出了常用的 HTML 服务器控件。

表 5.9　HTML 服务器控件

控　件	描　述	对应的 HTML 标记
HtmlAnchor	用于设置超级链接	<a>
HtmlButton	用于创建普通按钮	<button>
HtmlSelect	用于创建下拉列表	<select>
HtmlTextArea	用于创建多行文本框	<textarea>
HtmlInputButton	用于创建普通按钮	<input type="button">
HtmlInputCheckBox	用于创建复选框	<input type="check">

续表

控 件	描 述	对应的 HTML 标记
HtmlInputRadioButton	用于创建单选按钮	<input type="radio">
HtmlInputText	用于创建文本输入框	<input type="text">和<input type="password">
HtmlInputHidden	用于创建隐藏域	<input type="hidden">
HtmlInputImage	用于在页面上显示图片	<input type="image">
HtmlInputFile	用于上传文件	<input type="file">
HtmlForm	用于创建一个表单容器	<form>
HtmlImage	用于在页面上显示图片	<img>
HtmlTable	用于创建一个表格	<table>
HtmlTableRow	用于创建表格的一行	<tr>
HtmlTableCell	用于创建表格的一个单元格	<td>
HtmlGenericControl	用于执行那些不直接表现出来的未知的 HTML 控件	任何其他没有对应控件的标记，如<span>、<div>等

由于针对每一个 HTML 服务器控件，均会有一个 HTML 标记与其对应，因此在将 ASP 页面转为 ASP.NET 页面时，HTML 服务器控件将会显得尤为有用。也正是由于这一点，使得 HTML 服务器控件不具备任何抽象能力，即无法扩展其功能。为此，微软推出了适用于 ASP.NET 的一类新的服务器控件，这就是 Web 服务器控件。与 HTML 服务器控件相比，Web 服务器控件的功能更强大，使用更灵活。

Web 服务器控件位于 System.Web.UI.WebControls 命名空间中，是从 WebControl 基类中直接或间接派生出来的。与 HTML 控件一样，Web 服务器控件必须包含“Runat="Server"”属性，同时 Web 服务器控件也必须驻留在具有“Runat="Server"”属性的<Form>标记中。

Web 服务器控件的语法与 HTML 服务器的语法较为相似，其主要的区别在于 Web 服务器控件必须带有标记前缀“asp:”，如以下是一个 Text 服务器控件的声明：

```
<asp:TextBox ID="TextBox1" Text="" runat="server" ></asp:TextBox>
```

前缀“asp: ”用于将标记映射到运行时组件的命名空间，即 System.Web.UI.WebControls 命名空间，而标记的其余部分则是运行时类自身的名称。

下面，对 Web 服务器控件按功能进行分类介绍，并简要叙述各个控件的使用、声明示例以及在浏览器中的显示效果。

1. 文本显示控件

文本显示控件主要用于在页面上显示静态的文本，它包括 Label 控件和 Literal 控件。

1) Label 控件

Label 控件可以通过编程来动态地控制文本的显示内容，一般在当希望程序运行时动态改变页面中的文本时使用此控件。

声明示例：

```
<asp:Label Id="Label1" Text="显示文本"  Runat="Server" ></asp:Label>
```

2)　Literal 控件

Literal 控件与 Label 控件类似，也用于在页面上显示静态的文本。但与 Label 控件不同的是，Literal 控件不允许对所显示的文本应用样式。

声明示例：

```
<asp:Literal id="Literal1"  Text="显示文本"  Runat="Server" />
```

2. 输入控件

输入控件是指允许用户输入文本信息的控件，这里主要指 TextBox 控件。

TextBox 控件可以创建三种形式的文本框：单行文本框、多行文本框和密码文本框，其属性 TextMode 决定了当前所显示的文本框的形式。当属性 TextMode 的值为 Single 时，显示为单行文本框；当其值为 Multiline 时，显示为多行文本框；当其值为 Password 时，显示为密码文本框。

声明示例：

```
<asp:TextBox ID="TextBox1" runat="server" Text="单行文本" TextMode="
SingleLine" />
```

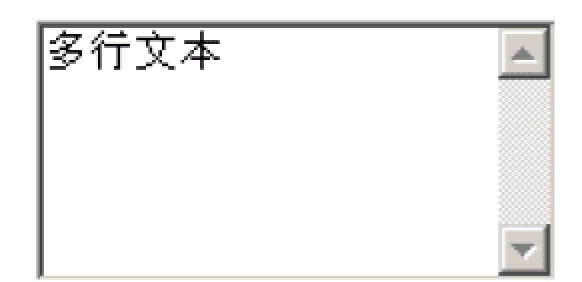
显示效果：

单行文本

声明示例：

```
<asp:TextBox ID="TextBox1" Rows="5" runat="server" Text="多行文本"
TextMode="MultiLine" />
```

显示效果：



声明示例：

```
<asp:TextBox ID="TextBox1" runat="server" TextMode="Password" />
```

显示效果：

****

3. 选择控件

选择控件是指在页面中提供相关选项，允许用户进行选择操作的控件。

1)　CheckBox 控件

CheckBox 控件主要用于创建能够在选中和清除两种状态间进行切换的复选框。通过其

Checked 属性，可以获取或设置当前复选框的状态。

声明示例：

```
<asp:CheckBox ID="Check1" runat="server" Checked="true" Text="CheckBox" />
```

显示效果：

☑ CheckBox

2)　CheckBoxList 控件

CheckBoxList 控件主要用于创建一组复选框控件，它使得创建含有多个 CheckBox 控件的页面更为简单。由于 CheckBoxList 控件是允许多项选择的，因此在判断用户所选择的项时必须通过一个循环来遍历所有条目，并判断该条目是否选择。

声明示例：

```
<asp:CheckBoxList id="Check1" RepeatDirection="Horizontal"
RepeatColumns=4 RepeatLayout="table" CellPadding="3" Runat="Server">
  <asp:ListItem>Check1</asp:ListItem>
  <asp:ListItem>Check2</asp:ListItem>
  <asp:ListItem>Check3</asp:ListItem>
  <asp:ListItem>Check4</asp:ListItem>
</asp:CheckBoxList>
```

显示效果：

☐ Check1 ☑ Check2 ☐ Check3 ☐ Check4

3)　DropDownList 控件

DropDownList 控件主要用于创建单一的下拉列表框，在其下拉列表框中可以包含任意数目的项。通过 SelectedIndex 属性，可以获取用户在下拉列表框中所选择的项的索引。

声明示例：

```
<asp:DropDownList ID="DropDownList1" runat="server">
  <asp:ListItem Value="Red" Selected="true">红　色</asp:ListItem>
  <asp:ListItem Value="Green">绿　色</asp:ListItem>
  <asp:ListItem Value="Black">黑　色</asp:ListItem>
</asp:DropDownList>
```

显示效果：

红 色 ▼

4)　ListBox 控件

ListBox 控件主要用于创建可滚动的列表框控件，并允许选择其中的一项或多个选项。通过 Rows 属性，可以设置 ListBox 控件所显示的行数；通过 SelectionMode 属性，可以设置 ListBox 控件的选择模式是单选(Single)还是多选(Multiple)；通过 SelectedItem 属性，可以获取当前所选择的项。

声明示例：

```
<asp:ListBox ID="ListBox1" Rows="5" runat="server" SelectionMode="
Multiple">
```

```
  <asp:ListItem Value="Red" Selected="true">红　色</asp:ListItem>
  <asp:ListItem Value="Green">绿　色</asp:ListItem>
  <asp:ListItem Value="Black">黑　色</asp:ListItem>
</asp:ListBox>
```

显示效果：

5)　RadioButton 控件

RadioButton 控件主要用于创建单个单选按钮。通过 Checked 属性，可以设置或获取单选按钮是否被选中。通过设置多个 RadioButton 控件的 GroupName 属性为同一个值，可创建一组互相排斥的选项，即一次只能选中组内的一个选项，而不能多选。

声明示例：

```
<asp:RadioButton ID="RadioButton1" runat="server" Checked="true" Text="单选按钮" />
```

显示效果：

单选按钮

6)　RadioButtonList 控件

RadioButtonList 控件用于创建一组互斥的单选按钮。相对于 RadioButton 控件来说，如果需要使用数据绑定来创建一组单选按钮，则 RadioButtonList 控件更易于使用，但单个 RadioButton 控件则更易于布局。

声明示例：

```
<asp:RadioButtonList ID="RadioButtonList1" CellPadding="3"
runat="server">
  <asp:ListItem Value="Radio1" Selected="true">单选按钮 1</asp:ListItem>
  <asp:ListItem Value="Radio2">单选按钮 2</asp:ListItem>
  <asp:ListItem Value="Radio3">单选按钮 3</asp:ListItem>
</asp:RadioButtonList>
```

显示效果：

单选按钮1
单选按钮2
单选按钮3

4. 图像控件

图像控件主要用于在页面中显示图像。在 ASP.NET 中提供了两种图像控件：Image 控件和 ImageMap 控件。

1)　Image 控件

Image 控件是个单纯的图像控件，其主要作用便是在页面上显示指定的图像。

声明示例：

```
<asp:Image id="Image1" ImageUrl="Image/Net.Gif" AlternateText="图像提示文本"
Runat="Server" />
```

显示效果：

2)　ImageMap 控件

ImageMap 控件是 ASP.NET 2.0 中新增的控件，又叫图片地图控件。ImageMap 控件可创建包含定义的作用点区域的图像，也就是说，可以在一副图片中设置很多的热区，当用户单击不同热区的时候可实现不同的响应，既可以让用户通过单击热区跳转到不同的 URL，又可以让用户通过单击热区运行不同的服务器代码。

声明示例：

```
<asp:ImageMap ID="ImageMap1" runat="server" Height="42px" ImageUrl="
Image/map.jpg"  Width="312px">
  <asp:RectangleHotSpot AlternateText="网易" Bottom="42" HotSpotMode="
PostBack" PostBackValue="163"  Right="111" />
  <asp:RectangleHotSpot AlternateText="新浪" Bottom="42" HotSpotMode="
PostBack" Left="111"  PostBackValue="sina" Right="229" />
  <asp:RectangleHotSpot AlternateText="GOOGLE 搜索" Bottom="42" HotSpotMode
="PostBack" Left="229" PostBackValue="google" Right="312" />
</asp:ImageMap>
```

显示效果：

5. 表单提交控件

表单提交控件是指用于提交页面数据至服务器的相关控件，包括 Button 控件、ImageButton 控件和 LinkButton 控件。

1)　Button 控件

Button 控件主要用于创建普通按钮。

声明示例：

```
<asp:Button ID="Button1" runat="server" Text="提 交" OnClick="Btn_Click" />
```

显示效果：

提　交

2)　ImageButton 控件

ImageButton 控件主要用于创建图形按钮，它结合了 Image 控件和 Button 控件，以 Image 控件的形式显示，而执行的是 Button 控件的操作。

声明示例：

```
<asp:ImageButton id="ImageButton1" AlternateText="图像按钮"
ImageAlign="Middle" ImageUrl=" Image/Net.Gif "
OnClick="ImageButton_Click" Runat="Server"/>
```

显示效果：

3) LinkButton 控件

LinkButton 控件主要用于创建超级链接样式的按钮，它结合了 HyperLink 控件和 Button 控件，具有与 HyperLink 控件完全相同的外观，而执行的操作与 Button 控件完全相同。

声明示例：

```
<asp:LinkButton id="LinkButton1" Text="提  交"
OnClick="LinkButton_Click"Runat="Server" />
```

显示效果：

提 交

6. 布局控件

布局控件主要用于控制页面的版面布局，其控件本身并不包含特定的功能。在 ASP.NET 中，布局控件包括 BulletedList 控件、MultiView 控件、Panel 控件、PlaceHolder 控件以及 Table 控件等。

1) BulletedList 控件

BulletedList 控件是 ASP.NET 2.0 中新增的一个控件，它可以在页面上轻松地创建项目符号和编号格式。过去，如果需要动态地显示项目符号和编号格式，要么使用 HTML 中的 或<ul>元素，要么使用 Repeater 控件。但前者过于死板，而后者则可谓是“杀鸡用牛刀”。

在 BulletedList 控件中，BulletStyle 属性和 DisplayMode 属性是最为重要的两个属性。其中，BulletStyle 属性表示项目符号的编号样式，如 Numbered(数字格式)、LowerAlpha(小写字母格式)、UpperAlpha(大写字母格式)、LowerRoman(小写阿拉伯数字)、UpperRoman(大写阿拉伯数字)、Disc(实心圆圈)、Circle(空心圆圈)以及 Square(实体黑方块)等。DisplayMode 属性则表示选项的显示模式，包括 Text(纯文本)、HyperLink(超链接)和 LinkButton(链接按钮)等。

声明示例：

```
<asp:BulletedList ID="BulletedList1" Font-Size="13px" runat="server"
BulletStyle="Square">
  <asp:ListItem Value="Item1">列表项目 1</asp:ListItem>
  <asp:ListItem Value="Item2">列表项目 2</asp:ListItem>
  <asp:ListItem Selected="True" Value="Item3">列表项目 3</asp:ListItem>
  <asp:ListItem Value="Item4">列表项目 4</asp:ListItem>
</asp:BulletedList>
```

显示效果：

- 列表项目1
- 列表项目2
- 列表项目3
- 列表项目4

2)　MultiView 控件

MultiView 控件也是 ASP.NET 2.0 中新增的控件，又称为多页面控件。MultiView 控件主要用来创建包含多个逻辑视图的页面，其中每个视图页面均对应一个 View 控件。View 控件是一个 Web 控件的容器，在其中可以添加其他的文本或控件，而 MultiView 控件则是为了显示 View 控件而定制的工具。从某种程度上来说，MultiView 控件有点类似于 C/S 开发中常见的 TabControl 控件，可以在一个页面上通过切换不同的选项卡来显示不同的内容。

通过设置 MultiView 控件的 ActiveViewIndex 属性，可以控制当前所显示的视图页面。

声明示例：

```
<asp:RadioButtonList ID="RadioButtonList1" CellPadding="3"
RepeatLayout="Table" runat="server" RepeatDirection="Horizontal">
  <asp:ListItem Value="Item1" Selected="true">视图页面</asp:ListItem>
  <asp:ListItem Value="Item2">视图页面</asp:ListItem>
</asp:RadioButtonList>
<asp:MultiView ID="MultiView1" ActiveViewIndex="0" runat="server">
  <asp:View ID="View1" runat="server">
    <p align=center>
      <asp:Image ID="Image1" ImageUrl="image/1.jpg" runat="server" />
      <br /><font color="red">这是第一个视图页面</font>
    </p>
  </asp:View>
  <asp:View ID="View2" runat="server">
    <p align=center>
      <asp:Image ID="Image2" ImageUrl="image/2.jpg" runat="server" />
      <br /><font color="red">这是第二个视图页面</font>
    </p>
  </asp:View>
</asp:MultiView>
```

显示效果：

3)　Panel 控件

Panel 控件是一种容器控件，它主要用于以编程的方式来生成多个控件，动态隐藏或显示一组控件。

声明示例：

```
<asp:Panel id="Panel1" runat="server" BackColor="gainsboro" Height="150px"
Width="200px" >
  <p align="center"><br />
  <asp:Label ID="Label1" runat="server" ForeColor="red" Font-Size="14px"
Text="这是在 Panel 控件里"></asp:Label><br /><br />
  <asp:Button ID="Button1" runat="server" Text="确 定" /></p>
</asp:Panel>
```

显示效果：

4) PlaceHolder 控件

PlaceHolder 控件也是一个容器控件，可以动态地添加各种 Web 控件。但是，与 Panel 控件相比，它并不产生任何可见的输出，而仅仅是作为一种容器存在。

声明示例：

```
<asp:PlaceHolder id="PlaceHolder1" runat="server">
  <p align="center"><br />
  <asp:Label ID="Label1" runat="server" ForeColor="red" Font-Size="14px"
Text="这是在 PlaceHolder 控件里"></asp:Label><br /><br />
  <asp:Button ID="Button1" runat="server" Text="确 定" /></p>
</asp:PlaceHolder>
```

5) Table 控件

Table 控件主要用于在页面上创建表格。与 HTMLTable 不同的是，Table 控件与相关的 TableRow 控件和 TableCell 控件相结合，可以通过编程的方式来动态地创建表格或控制版面布局。

声明示例：

```
<asp:Table ID="Table1" runat="server" GridLines="Both" Font-Size="13px">
  <asp:TableHeaderRow>
    <asp:TableHeaderCell>标题一</asp:TableHeaderCell>
    <asp:TableHeaderCell>标题二</asp:TableHeaderCell>
    <asp:TableHeaderCell>标题三</asp:TableHeaderCell>
  </asp:TableHeaderRow>
  <asp:TableRow>
    <asp:TableCell>[0,0]</asp:TableCell>
    <asp:TableCell>[0,1]</asp:TableCell>
    <asp:TableCell>[0,2]</asp:TableCell>
  </asp:TableRow>
```

```
  <asp:TableRow>
    <asp:TableCell>[1,0]</asp:TableCell>
    <asp:TableCell>[1,1]</asp:TableCell>
    <asp:TableCell>[1,2]</asp:TableCell>
  </asp:TableRow>
  <asp:TableRow>
    <asp:TableCell>[2,0]</asp:TableCell>
    <asp:TableCell>[2,1]</asp:TableCell>
    <asp:TableCell>[2,2]</asp:TableCell>
  </asp:TableRow>
</asp:Table>
```

显示效果：

标题一	标题二	标题三
[0, 0]	[0, 1]	[0, 2]
[1, 0]	[1, 1]	[1, 2]
[2, 0]	[2, 1]	[2, 2]

7. 导航控件

导航控件主要用于实现页面或站点的导航功能，该类控件包括 HyperLink 控件和 TreeView 控件。

1) HyperLink 控件

HyperLink 控件主要用于创建文本或图像的超级链接。通过属性 NavigateUrl，可以设置或获取链接的 URL 地址。同时，HyperLink 控件还提供了 Text 属性和 ImageUrl 属性，前者表示创建文本链接，后者表示创建图像链接。

声明示例：

```
文本链接：
<asp:HyperLink ID="HyperLink1" Text="链接文本" NavigateUrl="test1.aspx"
runat="server"></asp:HyperLink>
<p>图像链接：<br />
<asp:HyperLink ID="HyperLink2" ImageUrl="image/2.jpg" NavigateUrl="test1.
aspx" runat="server">图像链接</asp:HyperLink></p>
```

显示效果：

2) TreeView 控件

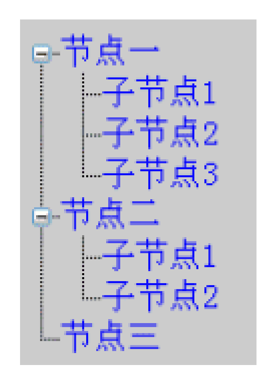
TreeView 控件是 ASP.NET 2.0 中新增的服务器控件，它主要用于创建树形视图，通常

可用来实现应用程序的功能菜单。在 ASP.NET 1.0 中，微软并未将 TreeView 控件作为 ASP.NET 的一个内置服务器控件，而是将其作为一个附加的 Web 控件集(即 Internet Explorer Web)中的一部分。

声明示例：

```
<asp:TreeView ID="TreeView1" runat="server" ExpandDepth="1"
ShowLines="True" ForeColor="Blue">
  <Nodes>
    <asp:TreeNode SelectAction="Expand" Text="节点一">
      <asp:TreeNode NavigateUrl="Page1.aspx" Target="main" Text="子节点 1">
      </asp:TreeNode>
      <asp:TreeNode NavigateUrl="Page2.aspx" Target="main" Text="子节点 2">
      </asp:TreeNode>
      <asp:TreeNode NavigateUrl="Page3.aspx" Target="main" Text="子节点 3">
      </asp:TreeNode>
    </asp:TreeNode>
    <asp:TreeNode SelectAction="Expand" Target="main" Text="节点二">
      <asp:TreeNode NavigateUrl="Page4.aspx" Target="main" Text="子节点 1">
      </asp:TreeNode>
      <asp:TreeNode NavigateUrl="Page5.aspx" Target="main" Text="子节点 2">
      </asp:TreeNode>
    </asp:TreeNode>
    <asp:TreeNode NavigateUrl="Page6.aspx" Target="main" Text="节点三">
    </asp:TreeNode>
  </Nodes>
</asp:TreeView>
```

显示效果：



8. 广告控件

广告控件主要用于实现广告图片的随机显示，这里主要是指 AdRotator 服务器控件。

AdRotator 控件，又叫广告控制元件，其作用是在每次浏览包含该控制元件的网页时，随机显示不同的广告图片，同时还可为所显示的广告图片创建相应的链接。在 AdRotator 控件中，属性 AdvertisementFile 指定了包含要发布的广告图片信息的 XML 文件。

声明示例：

```
<Asp:Adrotator id="adrotator1" AdvertisementFile="adrotator.xml"
runat="server" />
```

显示效果：

9. 日期控件

日期控件主要用于提供对日历的查看及对日期的选择。

Calendar 控件是一个日历控制控件，可以用来显示日历、浏览日期并进行日期选择。Calendar 控件不支持绑定到数据源，但可修改各个日期单元格的内容和格式设置。

声明示例：

```
<asp:Calendar id="Calendar1"
     SelectionMode="DayWeekMonth"
     Font-Size="13px"
     PrevMonthText="上一月"
     NextMonthText="下一月"
     SelectWeekText="选择星期"
     SelectMonthText="选择整月"
     runat="server">
  <todaydaystyle font-bold="true" />
  <dayheaderstyle font-bold="true" />
  <othermonthdaystyle forecolor="gray" />
  <titlestyle backcolor="#3366ff" forecolor="white" font-bold="true" />
  <selecteddaystyle backcolor="#ffcc66" font-bold="true" />
  <nextprevstyle forecolor="white" font-size="13px" />
  <selectorstyle backcolor="#99ccff" forecolor="navy" font-size="12px" />
</asp:calendar>
```

显示效果：

上一月				2006年11月			下一月
选择整月	日	一	二	三	四	五	六
选择星期	29	30	31	1	2	3	4
选择星期	5	6	7	8	9	10	11
选择星期	12	13	14	15	**16**	17	18
选择星期	19	20	21	22	23	24	25
选择星期	26	27	28	29	30	1	2
选择星期	3	4	5	6	7	8	9

10. 数据控件

数据控件主要用于通过数据绑定以列表或其他的布局方式在页面上显示其中的数据。在 ASP.NET 中，数据控件主要包括 DataGrid 控件、DataList 控件、Repeater 控件以及 GridView 控件和 DetailsView 控件。其中，GridView 控件和 DetailsView 控件均是 ASP.NET 2.0 中新增的数据控件。

第 6 章将详细介绍各种常用数据控件的使用，这里不再赘述。

11. 验证控件

验证控件主要用于对用户在输入表单中的各输入控件中所输入的值进行验证，以判断其输入是否正确、合法。当验证失败时，将会自动显示验证控件中所设置的错误信息，同时阻止数据的提交。

1) CompareValidator 控件

CompareValidator 控件，又叫比较验证控件，主要用于将输入控件的值与指定的常数值或其他输入控件的值按照指定的比较运算符(>、<、=、<>、>=、<=等)进行比较，以判

断两个值是否匹配。此外，CompareValidator 控件还可用来判断用户所输入的值是否可以转换为其 Type 属性所指定的数据类型。

声明示例：

```
<asp:CompareValidator id="CompareValidator1"
     Runat="server"
     ErrorMessage="验证失败"
     ControlToCompare="TextBox2"
     ControlToValidate="TextBox1"></asp:CompareValidator>
```

2)　RangeValidator 控件

RangeValidator 控件，又叫范围比较验证控件，主要用于验证用户的输入是否在指定的范围内。

声明示例：

```
<asp:RangeValidator id="RangeValidator1"
     Runat="server"
     ErrorMessage="输入数值应在 1 和 31 之间"
     ControlToValidate="TextBox1"
     MinimumValue="1"
     MaximumValue="31"
     Type="Integer"></asp:RangeValidator>
```

3)　RequiredFieldValidator 控件

RequiredValidator 控件，又叫必须字段验证控件，主要用于对用户是否输入了数据进行验证，以保证强制性输入。

声明示例：

```
<asp:RequiredFieldValidator id="RequiredFieldValidator2"
     Runat="server"
     ErrorMessage="输入不能为空！"
     ControlToValidate="TextBox1" ></asp:RequiredFieldValidator>
```

4)　RegularExpressionValidator 控件

RegularExpressionValidator 控件，又叫模式匹配控件，主要用于检查用户所输入的数据是否与用户所设置的正则表达式定义的模式相匹配。该控件可以说是所有验证控件中最灵活、功能最强大的验证控件。

声明示例：

```
<asp:RegularExpressionValidator Runat="Server"
     Id="RegularExpressionValidator2"
     ErrorMessage="邮箱地址输入错误！"
     ControlToValidate="TextBox1"
     ValidationExpression="[\w-]+@[\w-]+\.(com|net|org|edu|mil)$">
</asp:RegularExpressionValidator>
```

5)　CustomValidator 控件

CustomValidator 控件，又叫自定义验证控件，它允许用户编写自己的验证逻辑(通常为一个自定义的客户端或服务器端的函数)来对用户所输入的数据进行验证。

```
<asp:CustomValidator Runat="Server"
     Id="CustomValidator1"
     ControlToValidate="TextBox1"
     ErrorMessage="请输入正确的内容！"
     OnServerValidate="ValidateData"></asp:CustomValidator>
```

6) ValidationSummary 控件

ValidationSummary 控件，又叫验证摘要控件，主要用于在页面的某个位置显示当前页上所有验证控件的错误摘要信息。

```
<asp:ValidationSummary Runat="Server"
     Id="ValidationSummary1"
     HeaderText="错误信息："></asp:ValidationSummary>
```

5.3 典型实例：创建个人求职表单

本实例，将介绍如何创建一个用于个人求职的输入表单。在表单中将充分应用 ASP.NET 中的各种常用的 Web 服务器控件，以帮助大家加深理解。

表单及控件的布局如图 5.2 所示。

图 5.2 输入表单

对于输入表单的版面布局控制，最常见的是通过表格来实现，本例也不例外。在这里，将个人简历信息分为基本资料、联系方式、个人爱好、教育情况、工作经历以及求职意向等几大块。在每一块中，又包括相关的多项信息。对每项信息的输入或选择，均通过 ASP.NET 中的 Web 服务器控件来实现。

在个人简历中，性别、婚姻状况以及求职类型 3 项都是需要在多个选项中选择且仅能选择一项，为此采用了 RadioButtonList 控件来实现，而没有采用单个的 RadioButton 控件。相对而言，在需要提供一组互斥的选项时，RadioButtonList 控件将比 RadioButton 控件更为快捷和方便。

对于个人爱好来说，需要提供多个选项且能进行多选，因此这里采用了 CheckBoxList 控件来实现。

下面，来看页面的代码实现。

【5-1.aspx】

```
<%@ Page Language="VB" ContentType="text/html" %>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<title>个人简历</title>
</head>
<body>
<form runat="server">
  <p align="center"><b>个人简历</b></p>
  <table align=center style="font-size:13px;" cellPadding=0 bgcolor="gray"
cellSpacing=1 width=700 border=0>
  <tr height=25 bgColor="#00cccc">
    <td colspan=4 style="color:white;padding-left:5px;"><b>基本资料
</b></td>
  </tr>
  <tr height=25 bgcolor="white">
    <td align="right">真实姓名：</td>
    <td colspan=3 style="padding-left:5px;"><asp:TextBox ID="Name"
runat="server"></asp:TextBox></td>
  </tr>
  <tr height=25 bgcolor="white">
    <td width="14%" align="right">性别：</td>
    <td width="36%" style="padding-left:5px;">
      <asp:RadioButtonList ID="RadioButtonList1" runat="server"
CellPadding="2" CellSpacing="2" RepeatDirection="Horizontal">
        <asp:ListItem Text="先生" Value="男" Selected=true></asp:ListItem>
        <asp:ListItem Text="女士" Value="女"></asp:ListItem>
      </asp:RadioButtonList>
    </td>
    <td width="12%" align="right">婚姻状况：</td>
    <td width="38%" style="padding-left:5px;">
      <asp:RadioButtonList ID="RadioButtonList2" runat="server"
CellPadding="2" CellSpacing="2" RepeatDirection="Horizontal">
        <asp:ListItem Text="未婚" Value="未" Selected=true></asp:ListItem>
        <asp:ListItem Text="已婚" Value="已"></asp:ListItem>
      </asp:RadioButtonList>
    </td>
  </tr>
  <tr height=25 bgcolor="white">
    <td align="right">身份证号：</td>
```

```
    <td style="padding-left:5px;">
      <asp:TextBox ID="sfz" runat="server"></asp:TextBox></td>
    <td align="right">出生日期：</td>
    <td style="padding-left:5px;">
      <asp:TextBox ID="Birth" runat="server"></asp:TextBox></td>
  </tr>
  <tr height=25 bgcolor="white">
    <td align="right">民族：</td>
    <td style="padding-left:5px;">
      <asp:DropDownList ID="mz" runat="server">
        <asp:ListItem Text="汉族" Value="汉族" Selected="true">
        </asp:ListItem>
        <asp:ListItem Text="畲族" Value="畲族"></asp:ListItem>
        <asp:ListItem Text="苗族" Value="苗族"></asp:ListItem>
        <asp:ListItem Text="回族" Value="回族"></asp:ListItem>
        <asp:ListItem Text="维吾尔族" Value="维吾尔族"></asp:ListItem>
        <asp:ListItem Text="其他少数民族" Value="其他少数民族"></asp:ListItem>
      </asp:DropDownList>
    </td>
    <td align="right">籍贯：</td>
    <td style="padding-left:5px;">
      <asp:TextBox ID="jg" runat="server"></asp:TextBox>
    </td>
  </tr>
  <tr height=25 bgcolor="white">
    <td align="right">身高：</td>
    <td style="padding-left:5px;">
      <asp:TextBox ID="sg" runat="server"></asp:TextBox> (厘米)
    </td>
    <td align="right">体重：</td>
    <td style="padding-left:5px;">
      <asp:TextBox ID="tz" runat="server"></asp:TextBox> (公斤)
    </td>
  </tr>
  <tr height=25 bgColor="#00cccc">
    <td colspan=4 style="color:white;padding-left:5px;"><b>联系方式
</b></td>
  </tr>
  <tr height=25 bgcolor="white">
    <td align="right">联系地址：</td>
    <td style="padding-left:5px;">
      <asp:TextBox ID="Addr" runat="server"></asp:TextBox>
    </td>
    <td align="right">邮政编码：</td>
    <td style="padding-left:5px;">
      <asp:TextBox ID="Zip" runat="server"></asp:TextBox>
    </td>
  </tr>
  <tr height=25 bgcolor="white">
    <td align="right">固定电话：</td>
    <td style="padding-left:5px;">
```

```
        <asp:TextBox ID="Tel1" runat="server"></asp:TextBox>
      </td>
      <td align="right">移动电话：</td>
      <td style="padding-left:5px;">
        <asp:TextBox ID="Tel2" runat="server"></asp:TextBox>
      </td>
    </tr>
    <tr height=25 bgcolor="white">
      <td align="right">E-MAIL：</td>
      <td style="padding-left:5px;">
        <asp:TextBox ID="Email" runat="server"></asp:TextBox>
      </td>
      <td align="right">个人主页：</td>
      <td style="padding-left:5px;">
        <asp:TextBox ID="HomePage" runat="server"></asp:TextBox>
      </td>
    </tr>
    <tr height=25 bgColor="#00cccc">
      <td colspan=4 style="color:white;padding-left:5px;"><b>个人爱好
  </b></td>
    </tr>
    <tr height=25 bgcolor="white">
      <td colspan=4 style="padding-left:20px;">
        <asp:CheckBoxList ID="xqah" CellPadding=5 runat="server"
  RepeatDirection="Horizontal">
          <asp:ListItem>电脑游戏</asp:ListItem>
          <asp:ListItem>软件编程</asp:ListItem>
          <asp:ListItem>体育运动</asp:ListItem>
          <asp:ListItem>棋牌娱乐</asp:ListItem>
          <asp:ListItem>读书学习</asp:ListItem>
          <asp:ListItem>其他</asp:ListItem>
        </asp:CheckBoxList>
      </td>
    </tr>
    <tr height=25 bgColor="#00cccc">
      <td colspan=4 style="color:white;padding-left:5px;"><b>教育情况
  </b></td>
    </tr>
    <tr height=25 bgcolor="white">
      <td align="right">最高学历：</td>
      <td style="padding-left:5px;">
        <asp:DropDownList ID="xl" runat="server">
         <asp:ListItem Text="本科" Value="本科"
  Selected="true"></asp:ListItem>
         <asp:ListItem Text="大专" Value="大专"></asp:ListItem>
         <asp:ListItem Text="中专" Value="中专"></asp:ListItem>
         <asp:ListItem Text="其他" Value="其他"></asp:ListItem>
        </asp:DropDownList>
      </td>
      <td align="right">所学专业：</td>
      <td style="padding-left:5px;">
```

```
        <asp:TextBox ID="zy" runat="server"></asp:TextBox>
      </td>
    </tr>
    <tr height=25 bgcolor="white">
      <td align="right">毕业院校：</td>
      <td style="padding-left:5px;">
        <asp:TextBox ID="byyx" runat="server"></asp:TextBox>
      </td>
      <td align="right">毕业时间：</td>
      <td style="padding-left:5px;">
        <asp:TextBox ID="bysj" runat="server"></asp:TextBox> (如：.7)
      </td>
    </tr>
    <tr height=25 bgColor="#00cccc">
      <td colspan=4 style="color:white;padding-left:5px;"><b>工作经历
  </b></td>
    </tr>
    <tr height=25 bgcolor="white">
      <td colspan=4 style="padding-left:5px;padding-bottom:3px;padding-top:
  3px;">
        <asp:TextBox ID="TextBox1" runat="server" TextMode="MultiLine"
  Width="585px" Rows="5"></asp:TextBox>
      </td>
    </tr>
    <tr height=25 bgColor="#00cccc">
    <td colspan=4  style="color:white;padding-left:5px;"><b>求职意向
  </b></td>
    </tr>
    <tr height=25 bgcolor="white">
      <td align="right">求职类型：</td>
      <td style="padding-left:5px;">
        <asp:RadioButtonList ID="qzlx" runat="server" CellPadding="2"
  CellSpacing="2" RepeatDirection="Horizontal">
          <asp:ListItem Text="全职" Value="全职" Selected=true></asp:ListItem>
          <asp:ListItem Text="兼职" Value="兼职"></asp:ListItem>
          <asp:ListItem Text="两者兼可" Value="两者兼可"></asp:ListItem>
        </asp:RadioButtonList>
      </td>
      <td align="right">求职岗位：</td>
      <td style="padding-left:5px;">
        <asp:TextBox ID="qzgw" runat="server"></asp:TextBox>
      </td>
    </tr>
    <tr height=25 bgcolor="white">
      <td align="right">月薪要求：</td>
      <td style="padding-left:5px;">
        <asp:DropDownList ID="yxyq" runat="server">
         <asp:ListItem Text="面议" Value="面议"
  Selected="true"></asp:ListItem>
         <asp:ListItem Text="1000 以下" Value="1000 以下"></asp:ListItem>
         <asp:ListItem Text="1000--2000" Value="1000--2000"></asp:ListItem>
```

```
       <asp:ListItem Text="2000--3000" Value="2000--3000"></asp:ListItem>
       <asp:ListItem Text="3000 以上" Value="3000 以上"></asp:ListItem>
      </asp:DropDownList>
    </td>
    <td align="right">到职日期：</td>
    <td style="padding-left:5px;">
      <asp:DropDownList ID="dzrq" runat="server">
       <asp:ListItem Text="随时" Value="随时"
Selected="true"></asp:ListItem>
       <asp:ListItem Text="一个星期" Value="一个星期"></asp:ListItem>
       <asp:ListItem Text="一个月" Value="一个月"></asp:ListItem>
       <asp:ListItem Text="三个月" Value="三个月"></asp:ListItem>
       <asp:ListItem Text="半年之内" Value="半年之内"></asp:ListItem>
      </asp:DropDownList>
    </td>
  </tr>
  <tr height=30 bgcolor="white">
    <td colspan=4 align="center">
      <asp:Button ID="Button1" runat="server" Text="提交" />
    </td>
  </tr>
  </table>
  </form>
</body>
</html>
```

在本例中，没有对数据的提交事件进行处理。一般来说，首先需要创建一个数据库，然后在其中添加一个包含以上信息字段的数据表，并在对数据的提交事件上将用户所输入的数据添加至数据库中。第 6 章将介绍如何在 ASP.NET 中访问和操作数据库，学完之后，相信读者就能自行添加提交按钮的事件代码了。

5.4　习　　题

(1) 了解和掌握 ASP.NET 中的各种服务器控件的使用。

(2) 了解 ASP.NET 中的验证控件的使用，并在个人简历实例中添加对相关输入控件的验证。

(3) 在学习完本书第 6 章后，完善个人简历实例，添加“提交”按钮的单击事件。

第 6 章　ADO.NET 与数据库的访问

在第 5 章中，对 ASP.NET 页面的基本结构及 VB.NET 脚本语言的相关语法进行了详细的叙述。本章，将向大家介绍 ASP.NET 中专门用来处理数据访问的组件——ADO.NET。

6.1　ADO.NET 简介

ADO.NET 是 ADO(Microsoft ActiveX Data Objects)的一个后继版本，其主要目的是为了在.NET 框架上更容易地创建分布式的、数据共享的应用程序。由于 ADO.NET 提供了到所有 OLE DB 数据源的接口，因此它为存取任何类型的数据均提供了一个框架。通过 ADO.NET，可以在.NET 应用程序中连接不同类型的数据源，并对数据进行检索、处理和更新等操作。

ADO.NET 为多层结构的分布式应用程序提供了一种断开式的数据访问。所谓断开式的数据访问，是指在从服务器获取所需的数据后，就断开与数据源的连接，而在本机的缓存中处理数据。这种方式的优势在于，它极大地节省了数据库服务器的资源占用，从而为建立具有高度伸缩性的应用程序提供了保障。

提示： 这里所指的伸缩性是一个系统性能的指标，用于衡量应用程序在获取更多资源(内存、处理器等)时的执行能力。

在 ADO.NET 中，.NET 数据提供程序和 DataSet 是两个核心的组成部分。

.NET 数据提供程序提供了对数据库的连接，它相当于 DataSet 与数据源之间的通信层，为连接、存取、检索数据源中的信息提供了所有的机制。.NET 数据提供程序可以识别并处理两种类型的数据源：SQL Server 和可以通过 OLE DB 进行访问的任何数据源。前者仅用于与 SQL Server 进行交互，它为 SQL Server 和 DataSet 之间的通信提供了所有的方法；后者则主要用于协调 DataSet 和 OLE DB 数据源之间的通信。在.NET 数据提供程序中，提供了 ADO.NET 中的 4 个核心对象：Connection(用于与数据库建立连接)、Command(用于执行具体的数据库操作)、DataReader(用于从数据源中获取只进且只读的数据流)和 DataAdapter(用于填充 DataSet 对象)。

DataSet 是 ADO.NET 的断开式结构的核心组件，其目的是实现独立于任何数据源的数据访问。DataSet 可以用于多种不同的数据源，用于 XML 数据，或用于管理应用程序本地的数据。可以将它看成是数据库在内存中的一个复制。正是由于 DataSet 的存在，才使得程序员在编写.NET 应用程序时可以忽略数据库之间的差异，从而获得一致的编程模型。对于 DataSet，还将在 6.2 节中进行详细的叙述。

6.2　ADO.NET 中的常用对象

下面，介绍 ADO.NET 中的几个常用对象，包括 Connection、Command、DataReader、DataSet、DataAdapter 和 DataView 等。它们是 ADO.NET 的核心组件。

6.2.1　Connection 对象

Connection 对象用于获取一个与特定数据源的连接，这是执行所有数据库操作的一个必要前提。在 ADO.NET 中，提供了两种 Connection 对象：SQLConnection 和 OleDbConnection，前者基于 SQL Server .NET 数据提供程序，后者基于 OLE DB .NET 数据提供程序。

下面介绍 Connection 对象的常用属性和方法。

1. 常用属性

- ConnectionString：用于获取或设置数据库的连接字符串。
- ConnectionTimeOut：用于获取或设置数据库连接的超时时间，其单位为 s，默认值为 15s。
- DataBase：用于获取当前连接所使用的数据库名称。
- DataSource：获取当前连接的数据源。对于 SQLConnection 对象，其返回的值是连接的 SQL Server 实例的名称；而对于 OleDbConnection 对象，其返回的值是数据源的路径及文件名。
- Provider：返回 OLE DB Provider 的名字，该属性仅适用于 OleDbConnection 对象。
- ServerVersion：返回关于所连接的数据库的版本信息。
- State：返回当前连接的状态，其值为一个 ConnectionState 对象。

2. 常用方法

- Open：用于打开与数据库的连接。
- ChangeDatabase：用于改变已打开的数据库。
- CreateCommand：用于创建并返回一个与该连接相关联的 Command 对象。
- Close：用于关闭与数据库的连接。
- Dispose：用于在显式释放对象时关闭数据库连接。

6.2.2　Command 对象

在 ADO.NET 中，Command 对象主要用于执行数据库的相关操作，包括查询、删除、更新等。Command 对象负责生成一个请求，并将其传递给数据源。如果有返回值，则 Command 对象将以 DataReader、标量值或参数的形式返回结果。 Command 对象在 ADO.NET 中是一个非常重要的组成部分，后面将要叙述的 DataReader 对象和 DataAdapter 对象均须通过 Command 对象从数据库中获取数据，甚至 DataReader 对象本身就是由 Command 对象的 ExecuteReader()方法来创建的。因此，掌握 Command 对象的使用方法尤

为重要。

下面介绍 Connection 对象的常用属性和方法。

1. 常用属性

- CommandText：设置或获取 Command 对象所要在数据源上执行的 SQL 语句或存储过程名。
- CommandType：表示 CommandText 值的类型，其可能的取值为 StoredProcedure(存储过程名)、TableDirect(返回所有列的表名)和 Text(SQL 文本命令)，其中 TableDirect 仅适用于 OLE DB。
- CommandTimeout：设置或获取 Command 命令执行并产生一个错误前等待的时间，类型为 int，单位为 s，默认值为 30s；如将其值设置为 0，则表示没有限制超时时间。
- Connection：设置或返回与该命令相关联的 Connection 对象。
- Parameters：一个 Parameter 对象的集合。可以通过向该集合中添加新的 Parameter 对象，在执行 Command 命令时向存储过程传递参数。
- Transaction：返回该命令所处的 Transaction 对象。对于 SQLCommand，返回的是 SQLTransaction 对象；对于 OleDbCommand，返回的是 OleDbTransaction 对象；默认值为 null。
- UpdateRowSource：返回命令影响的记录数。

2. 常用方法

- Cancel：用于试图取消所执行的 Command 命令。
- CreateParameter：用于创建一个 Parameter 对象的实例，该对象表示 Command 对象的参数。
- Dispose：用于释放命令对象所占用的资源。
- ExecuteNonQuery：用于执行指定的 SQL 语句，并返回受影响的记录数。
- ExecuteReader：用于执行指定的 SQL 语句，并返回一个 DataReader 对象。
- ExecuteScalar：用于执行指定的 SQL 语句，并获取查询所返回的结果集中的第一行的第一列的值，该方法将忽略额外的列或行。
- ExecuteXmlReader：如果 Command 对象的 CommandType 属性为合法的包含 FOR XML 子句的 SQL 语句时，可通过此方法来返回一个 XmlReader 对象，该方法仅支持 SQLCommand 对象。
- Prepare：用于创建 Command 对象的编译版本。
- ResetCommandTimeOut：用于将 Command 对象的 CommandTimeOut 属性重置为默认值。

6.2.3　DataReader 对象

DataReader 对象主要用于获取 Command 对象执行的结果，即记录集，该记录集是只向前、只读的数据流。要访问 DataReader 对象中的数据，只能使用该对象的 Read()方法。DataReader 对象的创建只能通过 Command 对象的 ExecuteReader()方法，而不能直接使用构

造函数。当创建一个 DataReader 对象时，其指针将指向首记录的前一个位置，每次调用 Read()方法后其指针自动后移一个，并返回一个布尔值表示是否已到了记录的末尾。

需要特别注意的是，DataReader 对象具有独占性。也就是说，当使用 DataReader 对象时，与它关联的 Connection 将只为它服务，此时不能对 Connection 执行任何其他操作(除 Close()方法之外)。因此，在使用完 DataReader 对象之后，切记要将其关闭之后再执行其他操作。

下面介绍 DataReader 对象的常用属性和方法。

1. 常用属性

- Depth：用于返回当前行的嵌套深度。
- FieldCount：用于返回当前所指行的列数。
- IsClosed：用于返回当前对象是否可关闭。
- RecordAffected：用于返回影响的记录个数。

2. 常用方法

- Close：用于关闭当前所打开的 DataReader 对象。
- GetBoolean：用于获取指定列的 Boolean 类型的值。
  GetByte：用于获取指定列的 Byte 类型的值。
- GetChar：用于获取指定列的 Char 类型的值。
- GetDataTypeName：用于获取指定列的数据类型。
- GetDateTime：用于获取指定列的 DateTime 对象形式的值。
- GetDecimal：用于获取指定列的 Decimal 类型的值。
- GetDouble：用于获取指定列的 Double 类型的值。
- GetFieldType：用于获取指定列的数据类型的 Type 对象表示。
- GetFloat：用于获取指定列的 Float 类型的值。
- GetInt16：用于获取指定列的 Short 类型的值。
- GetInt32：用于获取指定列的 Int 类型的值。
- GetInt64：用于获取指定列的 Long 类型的值。
- GetName：用于获取指定列的列名。
- GetString：用于获取指定列的 String 类型的值。
- GetType：用于获取当前对象(DataReader)的 Type 对象表示。
- GetValue：用于获取以本机格式表示的指定列的值。
- IsDbNull：用于获取一个值，该值表示列中是否包含不存在的或缺少的值。
- NextResult：用于在读取批处理 SQL 语句的结果时，移动到下一个结果。
- Read：用于从数据源中读取一个或多个记录集，并返回一个布尔值以表示记录集是否已到最后一条记录。

6.2.4　DataAdapter 对象

DataAdapter 对象与 DataReader 对象一样，用于获取 Command 对象所执行的结果。但是，DataAdapter 对象主要作为 DataSet 和数据源之间的桥梁，它获取 Command 对象执行

的结果集合，并将其填充 DataSet。

DataAdapter 对象除了用来向 DataSet 对象填充数据外，还可将对 DataSet 中的数据所做的修改更新至实际的数据库中。

DataAdapter 对象的常用属性和方法如下。

1. 常用属性

- SelectCommand：用于获取或设置用来从数据源中查询记录的 SQL 语句或存储过程。
- InsertCommand：用于获取或设置用来向数据源中插入记录的 SQL 语句或存储过程。
- DeleteCommand：用于获取或设置用来从数据源中删除记录的 SQL 语句或存储过程。
- UpdateCommand：用于获取或设置用来更新数据源中的记录的 SQL 语句或存储过程。

2. 常用方法

- Dispose：用于删除 DataAdapter 对象，并释放其占用的资源。
- Fill：用于将数据源中的数据填充至指定的 DataSet 对象中。
- FillSchema：用于将 DataTable 加入到指定的 DataSet 中，并配置表的架构以匹配数据源中的架构。
- GetFillParmeters：用于获取一个用于 SELECT 命令的由 DataParameter 对象组成的数组。
- Update：在 DataSet 对象中的数据有所改动后更新数据源，包括插入、删除、更新等行为。

6.2.5　DataSet 对象

DataSet 是 ADO.NET 的核心组件之一，它是从数据源中检索到的数据在内存中的一个缓存。与 DataReader 不同的是，DataSet 是一种断开数据库连接的记录集合，而 DataReader 离不开与数据源的连接，它与底层数据是紧密联系在一起的。这种区别也是 DataSet 对象的真正价值所在。

如果需要在不同的页面之间或对同一页面的多次请示使用相同的记录集，则完全可以创建一个 DataSet 对象来保存数据，并将其驻留在服务器的缓存中。DataSet 对象不仅在缓存中保存了其中的数据，它还保留了数据库的关系视图，包括一个或多个表以及表之间的关系、约束等。事实上，它就相当于一个完整的数据库。

DataTable 对象是 DataSet 中的一个非常重要的对象，就相当于一个数据库表，包含了 DataSet 中的所有数据。一个对应 DataTable 对象的数据表就是一堆数据行(DataRow)和列(DataColumn)的集合，同时它还包含了表的关系(DataRelation)、外键(ForeignKeyConstraint)以及约束(UniqueConstraint)等信息。

DataSet 对象的创建有两种方法：通过 DataAdapter 对象来填充和通过代码来自行创建。

DataSet 对象的常用属性和方法如下。

1. 常用属性

- CaseSentive：用于指示对 DataTable 中的字符串进行比较时是否区分大小写。
- DataSetName：用于返回当前 DataSet 对象的名称。

- DefaultViewManager：用于返回一个 DataViewManager，后者包含 DataSet 中的数据组成的定制视图。
- EnforceConstraints：用于指出在更新数据时是否应遵守约束规则。
- ExtendedProperties：用于获取一个包含自定义用户信息的 PropertyCollection 对象。
- HasErrors：用于返回一个值，该值表示当前 DataSet 中的记录是否存在错误。
- Locale：用于比较字符串的区域信息，它返回一个 CultureInfo 对象。
- Relations：用于返回一个表示 DataSet 的表之间的所有关系的 DataRelationCollection 对象。
- Tables：用于返回一个代表 DataSet 中所有表的 DataTableCollection 对象。

2. 常用方法

- AcceptChange：用于提交在加载 DataSet 或上次调用 AcceptChange 以来对 DataSet 所做的所有修改。
- Clear：用于删除 DataSet 中的所有记录。
- Clone：用于生成一个与当前 DataSet 相同且不包含数据的 DataSet 对象。
- Copy：用于生成一个与当前 DataSet 相同且包含数据的 DataSet 对象。
- GetChanges：用于生成一个只包含修改过的数据的 DataSet 对象。
- HasChange：用于返回一个标识当前 DataSet 中的数据是否进行了修改的值，这些修改包括新增的行、已删除的行或已修改的行。
- Merge：用于将指定的数据集合并到当前的 DataSet 或 DataTable 中。
- RejectChange：用于撤消自创建 DataSet 以来或上次调用 AcceptChanges 以来对 DataSet 所做的修改。

6.2.6 DataView 对象

DataView 对象类似于数据库中的视图，只是它所属的对象并不是实际的数据库，而是 DataSet。DataView 代表了一个 DataTable 的数据查看方式，它表示 DataSet 对象中数据的定制视图。一般来说，数据默认的查看方式是按照数据库的数据表中数据的排列顺序，以表格的形式显示出来。但是，对于用户而言，常常需要利用排序(Sort)、筛选(Filter)、查找(Search)等属性来定义不同的数据查看方式，包括所显示的数据和显示的排序方式，这正是 DataView 对象的价值所在。

但是，DataView 对象不同于 Select 查询语句所执行的结果，它只是静态地反映数据库中的数据。当 DataView 对象中的数据发生改变时，与其相应的源数据也会发生改变。

DataView 对象的创建方法有以下 3 种。

- 使用 DataView 对象的构造函数，通过传入已有的 DataTable 的参数来创建。
- 使用 DataTable 类的 DefaultView 方法来创建新的 DataView。
- 根据表之间的关系，例如父子关系，由父表的 DataView 来创建子表的 DataView。

DataView 对象的常用属性和方法如下。

1. 常用属性

- AllowDelete：用于设置是否允许删除 DataView 中的记录。

- AllowEdit：用于设置是否允许编辑 DataView 中的记录。
- AllowNew：用于设置是否允许通过 AddNew 方法在视图中添加一条新记录。
- ApplyDefaultSet：用于设置是否使用默认的排序方式。
- Item(Index 1)：用于返回 DataView 中指定的一条记录。
- RowFilter：用于设置筛选条件，以及可以在 DataView 中查看哪些行。
- RowStateFilter：用于获取或设置 DataView 中的行状态筛选器。
- Sort：用于获取或设置 DataView 中的一个或多个排序列及排序顺序。
- Table：用于获取或设置数据的源 DataTable。

2. 常用方法

- AddNew：用于在视图中添加一条新的记录。
- BeginInit：用于在运行时初始化窗体上所使用的或另一个组件所使用的 DataView 对象。
- Delete(index)：用于删除 DataView 中由 index 所指定的记录。
- Dispose：用于释放 DateView 对象的当前实例。
- EndInit：用于在运行时结束窗体上所使用的或另一个组件所使用的 DataView 对象。
- Find：用于在 DataView 中按照指定的排序关键字查找指定的记录。

6.3 使用 ADO.NET 对象访问数据库的两个实例

在 6.2 节中，介绍了 ADO.NET 中的几个常用对象。在本节中，将通过两个访问数据库的简单实例，来介绍这些对象在应用中的实际操作。

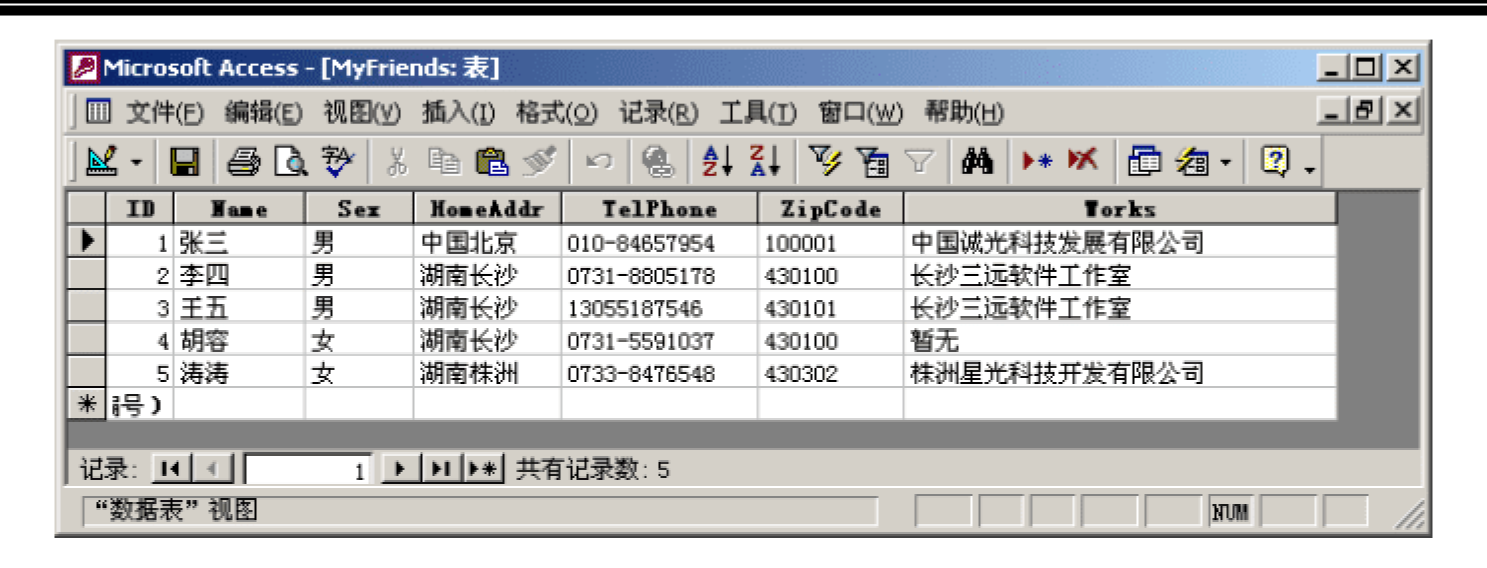
在介绍实例之前，先来创建示例数据库。在本章中，所有相关示例均把该数据库作为底层的数据库。

打开 Access，创建一个名为 Friend 的数据库，并在其中添加一个数据表 MyFriends，其表结构见表 6.1。

表 6.1 MyFriends 数据表

字 段 名	数据类型	字段描述	字段大小	默 认 值	主 键
ID	自动编号	序号	长整型		P
Name	文本	朋友姓名	50		
Sex	文本	性别	50		
HomeAddr	文本	家庭住址	100		
TelPhone	文本	联系电话	50		
ZipCode	文本	邮政编码	50		
Works	文本	工作单位	100		

双击该表，在表中添加相应的数据，如图 6.1 所示。



Microsoft Access - [MyFriends: 表]

ID	Name	Sex	HomeAddr	TelPhone	ZipCode	Works
1	张三	男	中国北京	010-84657954	100001	中国诚光科技发展有限公司
2	李四	男	湖南长沙	0731-8805178	430100	长沙三远软件工作室
3	王五	男	湖南长沙	13055187546	430101	长沙三远软件工作室
4	胡容	女	湖南长沙	0731-5591037	430100	暂无
5	涛涛	女	湖南株洲	0733-8476548	430302	株洲星光科技开发有限公司
号)						

记录: 1 共有记录数: 5

"数据表" 视图

图 6.1　示例数据

6.3.1　典型实例：使用 DataAdapter 对象访问数据

不同的数据提供程序使用的 DataAdapter 类也不同。如果访问 SQL Server 7.0 及以上版本的数据库，可使用 System.Data.SqlClient 命名空间中的 SqlDataAdapter 类；如果访问 OLE DB 接口的数据库，则须使用 System.Data.OleDb 命名空间中的 OleDbDataAdapter 类。这里，将使用后者。

首先，在 Dreamweaver 8 中新建一个 ASP.NET 页面，并将其保存在数据库 Friend 文件所在的文件夹中，命名为 UserDataAdapter.aspx。

然后，在页面中导入 System.Data 和 System.Data.OleDb 命名空间，如图 6.2 所示。

```
<html xmlns="http://www.w3.org/1999/xhtml">
<%@ Import NameSpace="System.Data" %>
<%@ Import NameSpace="System.Data.OleDb" %>
```

图 6.2　导入命名空间

并在代码中添加页面的 Page_Load 事件，在该事件中将实现数据的获取与绑定显示。

1. 使用 Connection 对象打开数据库

在查询数据之前，需要创建 Connection 对象，通过该对象来实现与指定数据库的连接，其代码如下：

```
Dim Cnn As OleDbConnection
Cnn=new OleDbConnection("Provider=Microsoft.Jet.OleDb.4.0;Data Source=" &
Server.MapPath("Friend.mdb"))
Cnn.open()
```

这里，定义了一个 OleDbConnection 对象，并在连接字符串中设置其连接的数据库为 Friend.mdb。然后，通过 Connection 对象的 Open()方法打开数据库连接。

2. 使用 DataAdapter 对象开启数据表

创建了数据库连接之后，则可使用 DataAdapter 对象来开启数据表。但是，DataAdapter 对象必须通过 Command 对象来实现，为此首先要创建一个 Command 对象，并设置其执行的 SQL 语句和关联的数据库连接。代码如下：

```
Dim Cmd As OleDbCommand
Dim dtCmd As OleDbDataAdapter
Cmd=New OleDbCommand("Select * from MyFriends order by id",Cnn)
dtCmd=New OleDbDataAdapter(Cmd)
```

该段代码定义了一个 Command 对象 Cmd 和一个 DataAdapter 对象 dtCmd，并设置了 Cmd 所执行的 SQL 语句与关联的数据库连接 Cnn，然后将 DataAdapter 对象实例化。自此，DataAdapter 对象已经创建成功，它已在 DataSet 对象和所需要查询的数据源之间搭起了一座“桥梁”。

3. 使用 DataSet 对象存放数据表

“桥梁”有了，接下来要做的就是通过 DataSet 对象来存放数据。代码如下：

```
Dim DS As DataSet
DS=New DataSet
dtCmd.Fill(DS,"friend")
```

在上面的代码中，定义了一个 DataSet 对象 DS，并进行了初始化操作。然后，通过 DataAdapter 对象的 Fill()方法将其数据填充至 DS 中的 Friend 表中。这里的 Friend 表并非固定的表，用户可自行定义其名称。此时，在 DataSet 中的 Friend 表中已经包含了用户所需数据。

4. 使用 DataGrid 控件显示数据表

将 DataSet 中的数据绑定在页面上的一个很简单的办法就是通过 DataGrid 控件来实现。DataGrid 控件是 ASP.NET 中用得最多的数据控件，对于它的基本语法，将在 6.4 节中进行叙述，这里不进行相关说明。

首先，在 ASP.NET 的<Body>标签中，添加一个 DataGrid 控件，其定义代码如下：

【定义 DataGrid 控件】

```
<asp:datagrid id="DataGrid1"  align=center runat="server" Width="544px"
AutoGenerateColumns="False" AllowPaging="False">
  <HeaderStyle BackColor="#D3F4FE" horizontalalign="center" font-size="11"
font-bold="true" forecolor="#660066" />
  <ItemStyle HorizontalAlign="Center" font-size="10" />
  <Columns>
    <asp:BoundColumn DataField="ID" HeaderText="编号"></asp:BoundColumn>
    <asp:BoundColumn DataField="Name" HeaderText="姓名"></asp:BoundColumn>
    <asp:BoundColumn DataField="Sex" HeaderText="性别"></asp:BoundColumn>
    <asp:BoundColumn DataField="HomeAddr" HeaderText="家庭住址">
    </asp:BoundColumn>
    <asp:BoundColumn DataField="TelPhone" HeaderText="联系电话">
    </asp:BoundColumn>
    <asp:BoundColumn DataField="Zipcode" HeaderText="邮政编码">
    </asp:BoundColumn>
    <asp:BoundColumn DataField="Works" HeaderText="工作单位">
    </asp:BoundColumn>
  </Columns>
</asp:datagrid>
```

然后，在 Page_Load 事件中，添加对 DataGrid 的绑定，代码如下：

```
DataGrid1.DataSource=DS.Tables("friend")
DataGrid1.DataBind()
Cnn.Close()
```

在上面的代码中，设置了 DataGrid 控件的数据源为 DataSet 中的数据表 Friend，然后通过 DataGrid 控件的 DataBind()方法实现与数据源的绑定。最后，别忘了调用 Connection 对象的 Close()方法来关闭打开的数据库连接。

至此，数据的获取和绑定显示完成，该页面的预览如图 6.3 所示。

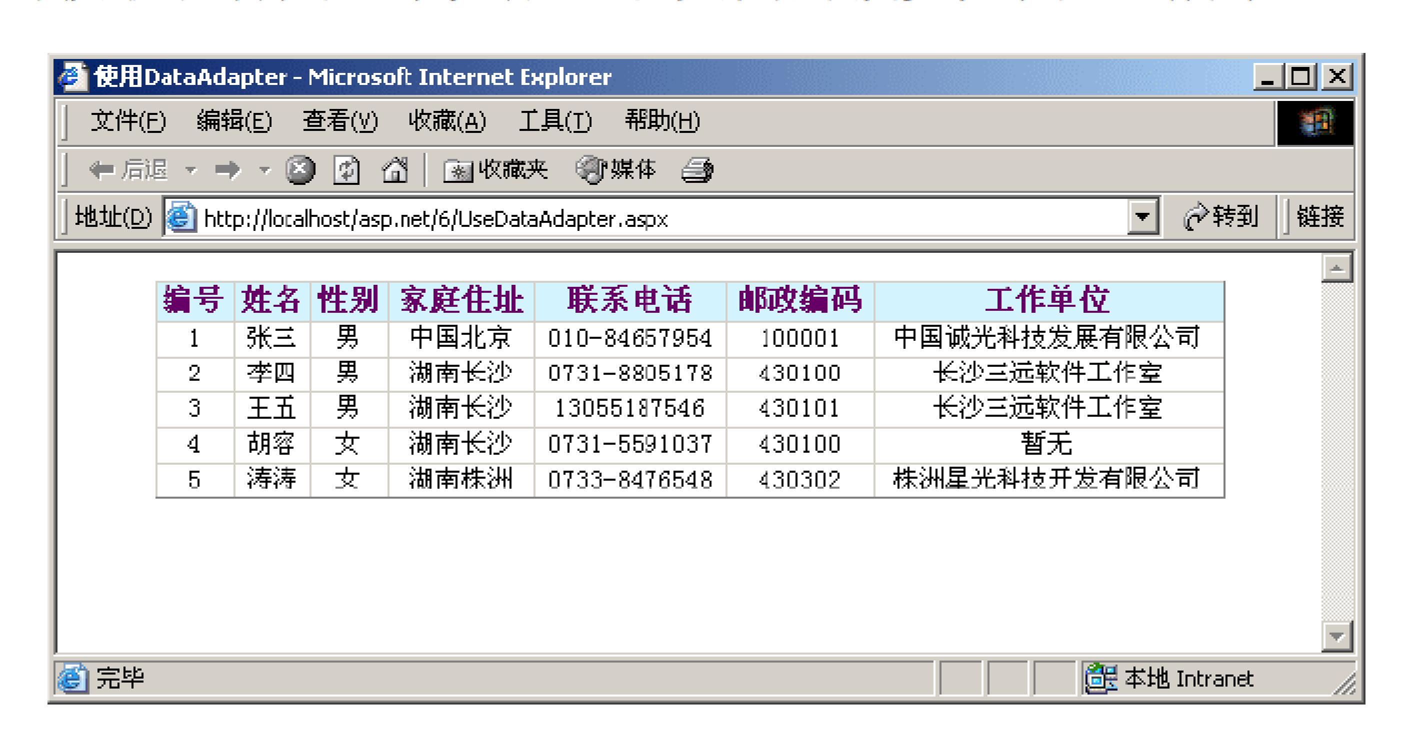


图 6.3　页面预览

最后，再来完整地看一下本页面的全部代码，如下：

【UseDataAdapter.aspx】

```
<%@ Page Language="VB" ContentType="text/html" ResponseEncoding="gb2312" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<%@ Import NameSpace="System.Data" %>
<%@ Import NameSpace="System.Data.OleDb" %>
<Script runat="server">
  Private Sub Page_Load(Byval Sender As Object,Byval E As EventArgs)
    Dim Cnn As OleDbConnection  '定义 Connection 对象
    Dim Cmd As OleDbCommand  '定义 Commnad 对象
    Dim dtCmd As OleDbDataAdapter  '定义 DataAdapter 对象
    Dim DS As DataSet  '定义 DataSet 对象
    If Not IsPostBack Then
      '创建数据库连接
      Cnn=new OleDbConnection("Provider=Microsoft.Jet.OleDb.4.0;Data
Source=" & Server.MapPath("Friend.mdb"))
      Cnn.open()
      '设置 Command 对象
      Cmd=New OleDbCommand("Select * from MyFriends order by id",Cnn)
      '设置 DataAdapter 对象
      dtCmd=New OleDbDataAdapter(Cmd)
      DS=New DataSet
      '将 DataAdapter 中的数据填充至 DataSet 中
      dtCmd.Fill(DS,"friend")
      '将 DataSet 绑定至 DataGrid 数据控件中
      DataGrid1.DataSource=DS.Tables("friend")
      DataGrid1.DataBind()
      Cnn.Close()  '关闭数据库连接
    End If
  End Sub
```

```
</Script>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<title>使用 DataAdapter</title>
</head>
<body>
<p align="center"><form id="Form1" runat="server">
<!--定义 DataGrid 控件-->
<asp:datagrid id="DataGrid1"  align=center runat="server" Width="544px"
AutoGenerateColumns="False" AllowPaging="False">
  <HeaderStyle BackColor="#D3F4FE" horizontalalign="center" font-size="11"
font-bold="true" forecolor="#660066" />
  <ItemStyle HorizontalAlign="Center" font-size="10" />
  <Columns>
    <asp:BoundColumn DataField="ID" HeaderText="编号"></asp:BoundColumn>
    <asp:BoundColumn DataField="Name" HeaderText="姓名"></asp:BoundColumn>
    <asp:BoundColumn DataField="Sex" HeaderText="性别"></asp:BoundColumn>
    <asp:BoundColumn DataField="HomeAddr" HeaderText="家庭住址">
    </asp:BoundColumn>
    <asp:BoundColumn DataField="TelPhone" HeaderText="联系电话">
    </asp:BoundColumn>
    <asp:BoundColumn DataField="Zipcode" HeaderText="邮政编码">
    </asp:BoundColumn>
    <asp:BoundColumn DataField="Works" HeaderText="工作单位">
    </asp:BoundColumn>
  </Columns>
</asp:datagrid>
<!--结束 DataGrid 控件的定义-->
</form></p>
</body>
</html>
```

6.3.2　典型实例：使用 DataReader 对象访问数据

前面介绍了 DataAdapter 对象的具体使用，接下来介绍如何使用 DataReader 对象来访问数据库中的数据。

对于 DataAdapter 对象，根据数据提供程序的不同，所使用的类也不同。对于 SQL Server，使用的类为 SqlDataAdapter；而对于 OLE DB，使用的类为 OleDbDataAdapter。

在 Dreamweaver 8 中新建一个 ASP.NET 页面，并将其保存在数据库 Friend 文件所在的文件夹中，命名为 UserDataReader.aspx。

然后，在页面中导入 System.Data 和 System.Data.OleDb 命名空间，并在代码中添加页面的 Page_Load 事件。

1. 使用 Connection 对象打开数据库

首先，创建用于建立数据库连接的 Connection 对象，此操作代码与前一实例中的连接数据的代码一样，如下：

```
Dim Cnn As OleDbConnection
Cnn=new OleDbConnection("Provider=Microsoft.Jet.OleDb.4.0;Data Source=" &
```

```
Server.MapPath("Friend.mdb"))
Cnn.open()
```

2. 使用 Command 对象开启数据表

接下来，创建 Command 对象，用于设置所要执行的查询 SQL 语句，同时绑定关联的 Connection 对象，代码如下：

```
Dim Cmd As OleDbCommand
Cmd=New OleDbCommand("Select * from MyFriends order by id",Cnn)
```

3. 使用 DataReader 对象连接数据表并读取数据表

然后，创建 DataReader 对象来获取 Command 对象执行所返回的记录集。

```
Dim DataR As OleDbDataReader
DataR=Cmd.ExecuteReader()
```

在以上代码中，定义了一个 OleDbDataReader 对象 DataR，并通过 Command 对象的 ExecuteReader()方法来实现初始化。此时，DataR 中已经包含了 Command 对象执行查询 SQL 语句所返回的记录集。

如前面所述，通过 DataReader 对象的 Read()方法可以访问其中的数据。下面，将通过一个调用 Read()方法的 While 循环来遍历 DataR 中的所有记录，并依次将其输出至页面上，代码如下：

```
While DataR.Read()
  Response.write("<p>")
  Response.Write (DataR("id") & "   " & DataR("Name") & "  " & DataR("Sex")
& " " & DataR("HomeAddr") & "   " & DataR("TelPhone") & "  " &
DataR("ZipCode") & "    " & DataR("Works"))
  Response.Write("</p>")
End While
```

需要注意的是，每执行一次 DataReader 对象的 Read()方法，其记录指针就自动后移一个，同时返回一个布尔值来表示是否已到了记录集的末尾。

由于页面数据的显示均是在代码中通过 Response.Write()方法输出，因此在页面的 <HTML>标记中无须添加任何文本或控件。

页面的预览效果如图 6.4 所示。

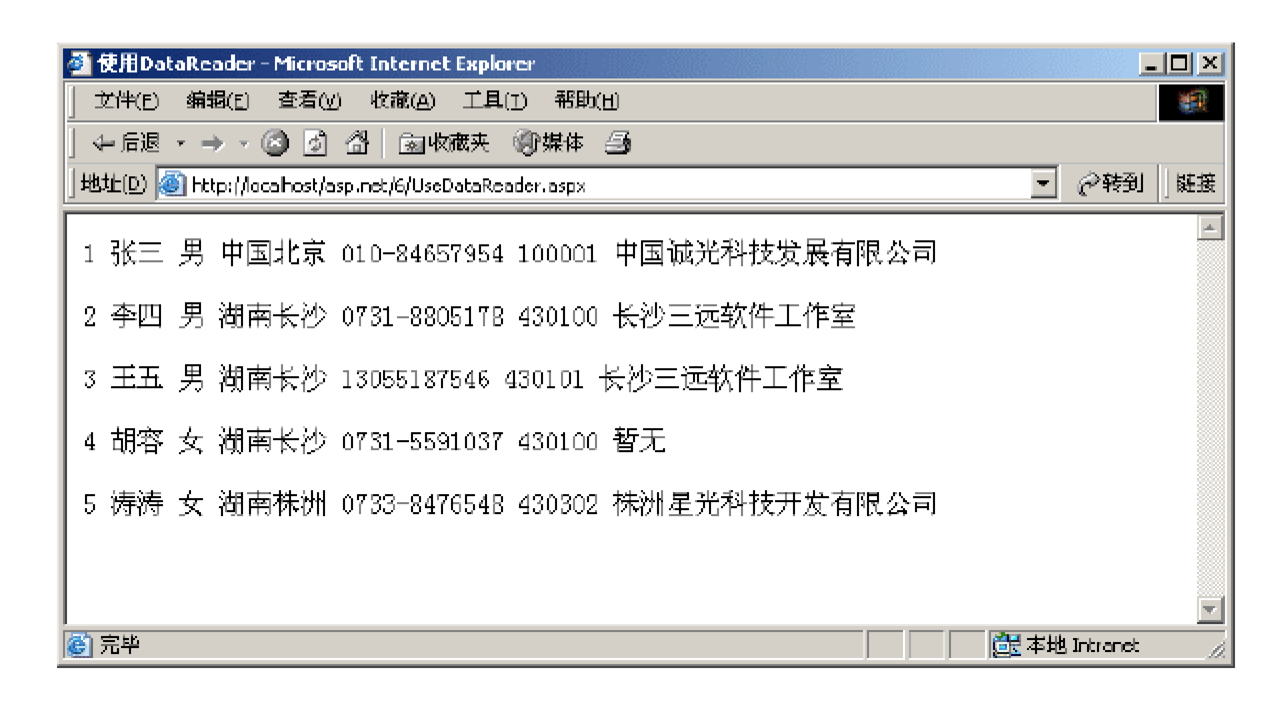


图 6.4　页面预览效果

本页面的完整代码如下所示：

【UseDataReader.aspx】

```
<%@ Page Language="VB" ContentType="text/html" ResponseEncoding="gb2312" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<%@ Import NameSpace="System.Data" %>
<%@ Import NameSpace="System.Data.OleDb" %>
<Script runat="server">
  Private Sub Page_Load(Byval Sender As Object,Byval E As EventArgs)
    Dim Cnn As OleDbConnection  '定义 Connection 对象
    Dim Cmd As OleDbCommand  '定义 Command 对象
    Dim DataR As OleDbDataReader  '定义 DataReader 对象
    If Not IsPostBack Then
      '创建数据库连接
      Cnn=new OleDbConnection("Provider=Microsoft.Jet.OleDb.4.0;Data
Source=" & Server.MapPath("Friend.mdb"))
      Cnn.open()
      '设置 Command 对象
      Cmd=New OleDbCommand("Select * from MyFriends order by id",Cnn)
      '设置 DataReader 对象
      DataR=Cmd.ExecuteReader()
      '遍历 DataReader 对象
      While DataR.Read()
        Response.write("<p>")
        '输出当前记录
        Response.Write (DataR("id") & " " & DataR("Name") & " " & DataR("Sex")
& " " & DataR("HomeAddr") & " " & DataR("TelPhone") &  " " & DataR("ZipCode")
&  " " & DataR("Works"))
        Response.Write("</p>")
      End While
      Cnn.Close()  '关闭数据库连接
    End If
  End Sub
</Script>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<title>使用 DataReader</title>
</head>
<body></body>
</html>
```

6.4　DataGrid 控件的使用

6.4.1　DataGrid 控件概述

DataGrid 控件是 ASP.NET 中用得最多的一种数据控件，它主要通过表格的布局方式来显示数据。对于其布局外观，用户可以进行灵活的定义和设置。默认情况下，DataGrid 以

只读的模式显示数据，但是用户也可设置其在运行时自动在可编辑控件中显示数据，并对其进行相应的编辑。

DataGrid 控件必须通过 DataSource 属性来绑定数据源，否则无法在页面上正常呈现。DataGrid 控件的数据源可以是 DataSet 或数据读取器。

默认情况下，DataGrid 控件中的列是自动生成的。但除此之外，DataGrid 控件还可以定义显示声明的列，以获得特殊的效果。在 DataGrid 控件中，可以定义 5 种显示声明的列，它们分别是 BoundColumn、ButtonColumn、EditCommandColumn、HyperLinkColumn 和 TemplateColumn。

其中，BoundColumn 为绑定列，它表示该列将绑定到数据源中的一个字段；ButtonColumn 为按钮列，它表示该列中的每一项均为一个用户自定义的按钮；EditCommandColumn 是一种特殊的列类型，它包含了用于编辑每行数据的各种编辑命令按钮，如编辑、更新、取消等；HyperLinkColumn 为超链接列，用于创建绑定到数据字段值的超级链接的列；TemplateColumn 为模板列，其所创建的列允许使用自定义 HTML 元素和控件的模板来定义控件的布局，该列也是所有显示声明的列中使用最为灵活、功能最为强大的列。

显式声明的列与自动生成的列可以同时存在。当两者同时使用时，先显示显式声明的列，再显示自动生成的列。对于自动生成的列，将不会添加到 Columns 集合中。在 6.3 节的第一个实例中所调用的 DataGrid 控件就是调用的显示声明的列。

DataGrid 控件的常用属性如下。

- DataSource：用于设置 DataGrid 控件的数据源。
- AllowPaging：表示是否启用分页功能，其每页显示的记录数由 PageSize 属性来指定。
- AllowSorting：表示是否启用排序功能。
- AutoGenerateColumns：表示是否为数据源中的每一列自动创建 BoundColumn 对象。
- BackImageUrl：用于设置 DataGrid 控件的背景图像。
- CellPadding：用于设置单元格的内容与单元格边框之间的距离。
- CellSpacing：用于设置单元格与单元格之间的距离。
- CurrentPageIndex：用于设置当前显示页的索引。
- DataKeyField：用于设置由 DataSource 属性指定的数据源中的主键字段。
- GridLines：表示是否显示 DataGrid 控件的单元格之间的边框。
- HorizontalAlign：表示 DataGrid 控件的水平对齐方式。
- PageSize：用于设置 DataGrid 控件每页所显示的项数，只有将 DataGrid 控件的 AllowPaging 属性设置为 True，该属性才有效。
- ShowFooter：表示是否在 DataGrid 控件中显示页脚。
- ShowHeader：表示是否在 DataGrid 控件中显示页眉。
- VirtualItemCount：用于设置自定义分页时 DataGrid 控件的实际项数。
- AlternatingItemStyle：用于设置 DataGrid 控件中交替项的样式。
- EditItemStyle：用于设置在 DataGrid 控件中选定来进行编辑的项的样式。

- FooterStyle：用于设置 DataGrid 控件中脚注部分的样式。
- HeaderStyle：用于设置 DataGrid 控件中标题部分的样式。
- ItemStyle：用于设置 DataGrid 控件中各项的样式。
- PageStyle：用于设置 DataGrid 控件中分页节的样式。
- SelectedItemStyle：用于设置 DataGrid 控件中当前选定项的样式。

6.4.2 DataGrid 控件的应用实例

在 6.4.1 节中，已经初步使用了 DataGrid 控件，但那仅仅是调用 DataGrid 控件来显示数据，并没有更深地扩展 DataGrid 控件的其他功能。

下面，将在 6.4.1 节的第一个实例的基础上，进一步添加 DataGrid 的其他强大功能，包括分页显示和对数据的编辑、删除等。

对于 DataGrid 的分页显示，首先需要设置其 AllowPaging 属性的值为 True，然后设置 DataGrid 控件的 PageSize 属性，该属性表示每一页所显示的记录数。接着需要分别设置 DataGrid 控件的 PagerStyle-PrevPageText 属性和 PagerStyle-NextPageText 属性。前者设置的文本代表上一页的标识，而后者所设置的文本则代表下一页的标识。最后，还需设置 DataGrid 控件的 OnPageIndexChanged 事件，该事件将在单击 DataGrid 中的【上一页】或【下一页】按钮时触发。一般来说，在该事件中需要进行两步操作：设置 DataGrid 控件的 CurrentPageIndex 属性，然后重新绑定数据。

对于 DataGrid 控件的编辑操作，首先需要在 DataGrid 中分别添加 EditCommandColumn 列和 ButtonColumn 列。前者用于显示编辑链接按钮，并执行相应的编辑操作；后者则用于显示删除按钮，并执行相应的删除操作。然后，需要分别添加 DataGrid 控件的 OnEditCommand 事件、OnCancelCommand 事件、OnUpdateCommand 事件和 OnItemCommand 事件。

其中，OnEditCommand 事件在用户单击 DataGrid 控件的编辑按钮时触发，其所执行的操作主要是设置 DataGrid 控件的 EditItemIndex 属性为当前单击的编辑按钮所在的行的索引。在执行此事件之后，编辑按钮将会转化为更新按钮和取消按钮，同时 DataGrid 控件中没有设置 ReadOnly 属性为 True 的所有字段列均将显示为一个文本编辑框，以便用户进行修改。OnCancelCommand 事件是在用户单击取消按钮时触发，其所执行的操作主要是 DataGrid 控件的 EditItemIndex 属性设置为-1，即取消编辑状态。OnUpdateCommand 事件是在用户单击更新按钮时触发，其所执行的操作是将用户对各文本框中的内容所作的修改提交至数据库中。OnItemCommand 事件，则主要针对于删除按钮，并在单击删除按钮时触发，其所执行的操作是从数据库中删除用户指定的数据。

下面，来看一下页面的程序代码：

【DataGrid.aspx】

```
<%@ Page Language="VB" ContentType="text/html" ResponseEncoding="gb2312" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<%@ Import NameSpace="System.Data" %>
<%@ Import NameSpace="System.Data.OleDb" %>
```

```
<Script runat="server">
  Private Sub Page_Load(Byval Sender As Object,Byval E As EventArgs)
    If Not IsPostBack Then
   '调用过程 BindData，将数据绑定至 DataGrid 控件中
      BindData()
    End If
  End Sub
  '自定义过程 BindData，用于从数据库中获取数据并将其绑定至 DataGrid 控件中
  Sub BindData()
    Dim Cnn As OleDbConnection  '定义 Connection 对象
    Dim Cmd As OleDbCommand  '定义 Command 对象
    Dim dtCmd As OleDbDataAdapter  '定义 DataAdapter 对象
    Dim DS As DataSet  '定义 DataSet 对象
    ' 创建数据库连接
    Cnn=new  OleDbConnection("Provider=Microsoft.Jet.OleDb.4.0;Data
Source=" & Server.MapPath("Friend.mdb"))
    Cnn.open()
    '获取数据
    Cmd=New OleDbCommand("Select * from MyFriends order by id",Cnn)
    dtCmd=New OleDbDataAdapter(Cmd)
    DS=New DataSet
    dtCmd.Fill(DS,"friend")
    '将数据绑定至 DataGrid 控件中
    DataGrid1.DataSource=DS.Tables("friend")
    DataGrid1.DataBind()
    Cnn.Close()  '关闭数据库连接
  End Sub
  '定义删除事件
  Sub DataGrid1_Mod(Sender as object,E as DataGridCommandEventArgs)
    Dim Cnn As OleDbConnection
    Dim Cmd As OleDbCommand
    '获取当前所要删除的记录的 ID
    dim Str_ID As String=E.Item.Cells(0).Text
    dim Sql as string
    If CType(e.CommandSource,LinkButton).CommandName="Delete" then
      '创建数据库连接
      Cnn=new OleDbConnection("Provider=Microsoft.Jet.OleDb.4.0;Data
Source=" & Server.MapPath("Friend.mdb"))
      Cnn.Open()
      '执行删除操作
      Sql="delete from MyFriends Where ID=" & Str_ID
      Cmd=New OleDbCommand(Sql,Cnn)
      Cmd.executeNonQuery
      Cnn.close()  '关闭数据库连接
      BindData()  '调用过程 BindData，重新绑定数据
    End if
  End Sub
  '定义更新事件
  Sub DataGrid1_Update(Sender as object,E as DataGridCommandEventArgs)
```

```
    Dim Cnn As OleDbConnection
    Dim Cmd As OleDbCommand
    '获取当前所要更新的记录的 ID
    Dim Str_ID As String=E.Item.Cells(0).Text
    '获取修改后的姓名列的值
    Dim Str_Name As String=CType(E.Item.Cells(1).Controls(0),TextBox).Text
    '获取修改后的性别列的值
    Dim Str_Sex As String=CType(E.Item.Cells(2).Controls(0),TextBox).Text
    '获取修改后家庭住址列的值
    Dim Str_Addr As String=CType(E.Item.Cells(3).Controls(0),TextBox).Text
    '获取修改后的联系电话列的值
    Dim Str_Tel As String=CType(E.Item.Cells(4).Controls(0),TextBox).Text
    '获取修改后的邮政编码列的值
    Dim Str_Zip As String=CType(E.Item.Cells(5).Controls(0),TextBox).Text
    '获取修改后的工作单位列的值
    Dim Str_Work As String=CType(E.Item.Cells(6).Controls(0),TextBox).Text
    Dim Sql As String
    '创建数据库连接
    Cnn=new OleDbConnection("Provider=Microsoft.Jet.OleDb.4.0;Data
Source=" & Server.MapPath("Friend.mdb"))
    Cnn.Open()
    '对数据库执行更新操作
    Sql="Update MyFriends Set Name='" & Trim(Str_Name) & "',Sex='" &
Trim(Str_Sex) & "',HomeAddr='" & Trim(Str_Addr) & "',TelPhone='" &
Trim(Str_Tel) & "',ZipCode='" & Trim(Str_Zip) & "',Works='" & Trim(Str_Work)
& "' Where ID=" & Trim(Str_ID)
    Cmd=New OleDbCommand(Sql,Cnn)
    Cmd.ExecuteNonQuery
    Cnn.Close()  '关闭数据库连接
    '将 DataGrid 的 EditItemIndex 属性置为-1，取消编辑状态
    DataGrid1.EditItemIndex=-1
    BindData()  '调用自定义过程 BindData，重新绑定数据
  End Sub
  '定义取消事件
  Sub DataGrid1_Cancel(Sender as object,E as DataGridCommandEventArgs)
    '将 DataGrid 的 EditItemIndex 属性置为-1，取消编辑状态
    DataGrid1.EditItemIndex=-1
    BindData()  '调用自定义过程 BindData，重新绑定数据
  End Sub
  '定义编辑事件
  Sub DataGrid1_Edit(Sender as object,E as DataGridCommandEventArgs)
    '设置要编辑的行的索引为当前所在的行的索引
    DataGrid1.EditItemIndex=E.Item.ItemIndex
    BindData()
  End Sub
  '定义分页事件
  Sub DataGrid1_PageIndexChanged(ByVal source As Object, ByVal e As
DataGridPageChangedEventArgs)
    '设置当前显示页的索引
```

```
    Datagrid1.CurrentPageIndex = e.NewPageIndex
    BindData()
  End Sub
</Script>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<title>使用 DataGrid 控件</title>
</head>
<body>
<p align="center">
<form id="Form1" runat="server">
<asp:Datagrid id="DataGrid1" align=center runat="server" Width="750px"
      AutoGenerateColumns="False"
      CellPadding="5"
      AllowPaging="True"
      PageSize="3"
      PagerStyle-NextPageText="下一页"
      PagerStyle-PrevPageText="上一页"
      OnPageIndexChanged="DataGrid1_PageIndexChanged"
      OnEditCommand="DataGrid1_Edit"
      OnCancelCommand="DataGrid1_Cancel"
      OnupdateCommand="DataGrid1_Update"
      OnItemCommand="DataGrid1_Mod" >
  <HeaderStyle BackColor="#D3F4FE" horizontalalign="center" font-size="11"
font-bold="true" forecolor="#660066" />
  <ItemStyle HorizontalAlign="Center" font-size="10" />
  <Columns>
    <asp:BoundColumn DataField="ID" HeaderText="编号" ReadOnly="True" >
    </asp:BoundColumn>
    <asp:BoundColumn DataField="Name" HeaderText="姓名"></asp:BoundColumn>
    <asp:BoundColumn DataField="Sex" HeaderText="性别"></asp:BoundColumn>
    <asp:BoundColumn DataField="HomeAddr" HeaderText="家庭住址">
    </asp:BoundColumn>
    <asp:BoundColumn DataField="TelPhone" HeaderText="联系电话">
    </asp:BoundColumn>
    <asp:BoundColumn DataField="Zipcode" HeaderText="邮政编码">
    </asp:BoundColumn>
    <asp:BoundColumn DataField="Works" HeaderText="工作单位">
    </asp:BoundColumn>
    <asp:EditCommandColumn ButtonType="LinkButton" UpdateText="更新"
HeaderText="编辑" CancelText="取消" EditText="编辑">
    </asp:EditCommandColumn>
    <asp:ButtonColumn Text="删除" HeaderText="删除" CommandName="Delete">
    </asp:ButtonColumn>
  </Columns>
</asp:datagrid>
</form></p>
</body>
</html>
```

页面预览效果如图 6.5 所示。

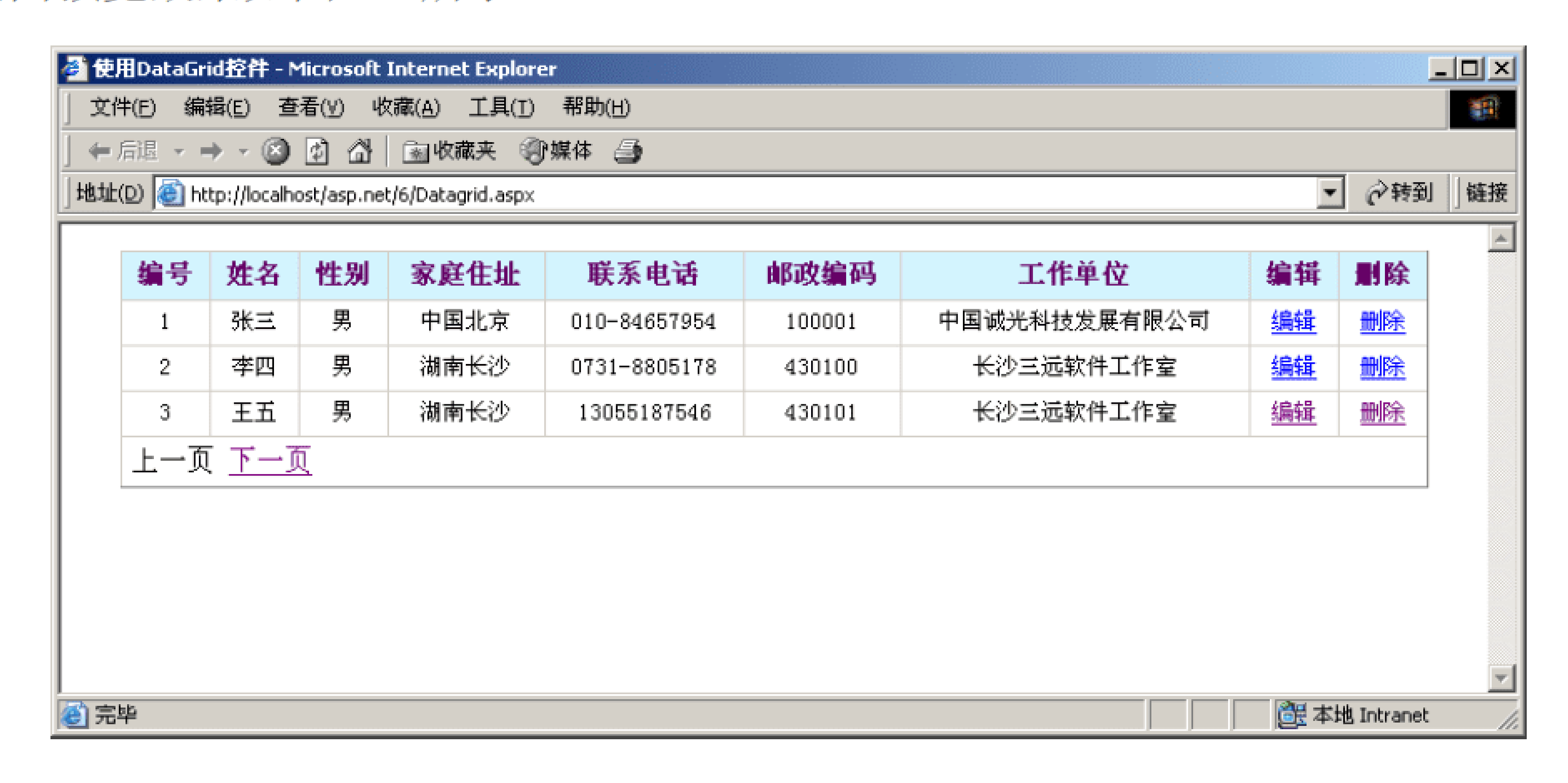


图 6.5　页面预览效果

6.5　DataList 控件和 Repeater 控件

DataList 控件和 Repeater 控件，也是 ASP.NET 中两个常用的数据控件。其中，DataList 控件主要用于通过模板来显示数据，而 Repeater 控件则可以采用更自由的方式来控制数据的显示。

6.5.1　DataList 控件

DataList 控件主要用于通过模板来显示数据。DataList 控件所显示的数据的格式，都可以通过模板或样式来定义。

在 DataList 控件所提供的模板中包括 AlternatingItemTemplate、EditItemTemplate、FooterTemplate、HeaderTemplate、ItemTemplate、SelectedItemTemplate、SeparatorTemplate 等。

其中，AlternatingItemTemplate 模板主要用于控制交替行所显示的文本和控件样式；EditItemTemplate 模板主要用于控制处于编辑状态下的项的文本和控件样式；FooterTemplate 模板主要用于控制 DataList 控件的脚注所显示的文本和控件样式；HeaderTemplate 模板主要用于控制 DataList 控件的标头所显示的文本和控件样式；ItemTemplate 模板主要用于控制 DataList 控件中的每一行所显示的文本和控件样式；SelectedItemTemplate 模板主要用于控制用户所选中的 DataList 控件的一项所显示的文本和控件样式；SeparatorTemplate 模板主要用于控制 DataList 控件中各项间分隔符的样式。

通过定义各种不同的模板，可以完全自定义 DataList 控件的外观样式和显示内容。

DataList 控件的常用属性如下。

- AlternatingItemStyle：用于获取 DataList 控件中交替项的样式属性。
- AlternatingItemTemplate：用于获取或设置 DataList 控件中交替项的模板。
- EditItemIndex：用于获取或设置 DataList 控件中所要编辑的选定项的索引号。
- EditItemStyle：用于获取 DataList 控件中为进行编辑而选定的项的样式属性。
- EditItemTemplate：用于获取或设置 DataList 控件中为进行编辑而选定的项的模板。

- FooterStyle：用于获取 DataList 控件的脚注部分的样式属性。
- FooterTemplate：用于获取或设置 DataList 控件的脚注部分的模板。
- GridLines：用于获取或设置 DataList 控件的网格线样式，此属性仅当 DataList 控件的 RepeatLayout 属性设置为 RepeatLayout.Table 时有效。
- HeaderStyle：用于获取 DataList 控件的标题部分的样式属性。
- HeaderTemplate：用于获取或设置 DataList 控件的标题部分的模板。
- Items：用于获取表示控件内单独项的 DataListItem 对象的集合。
- ItemStyle：用于获取 DataList 控件中项的样式属性。
- ItemTemplate：用于获取或设置 DataList 控件中项的模板。
- RepeatColumns：用于获取或设置要在 DataList 控件中显示的列的数量。
- RepeatDirection：用于获取或设置 DataList 控件的显示是垂直方向显示还是水平方向显示。
- RepeatLayout：用于获取或设置 DataList 控件是在表中显示还是在流布局中显示，其属性值包括 RepeatLayout.Table(以表的形式显示)和 RepeatLayout.Flow(以流布局的形式显示)。
- SelectedIndex：用于获取或设置 DataList 控件中选定项的索引。
- SelectedItem：用于获取 DataList 控件中的选定项。
- SelectedItemStyle：用于获取 DataList 控件中选定项的样式属性。
- SelectedItemTemplate：用于获取或设置 DataList 控件中选定项的模板。
- SeparatorStyle：用于获取 DataList 控件中各项间的分隔符的样式属性。
- SeparatorTemplate：用于获取或设置 DataList 控件中各项间的分隔符的模板。
- ShowFooter：用于获取或设置是否在 DataList 控件中显示脚注部分。
- ShowHeader：用于获取或设置是否在 DataList 控件中显示页眉节。

6.5.2　Repeater 控件

Repeater 控件可以使用非表格的形式来显示数据，从而能够非常灵活地定义其显示的风格。对于 Repeater 控件来说，它相当于一个基本容器控件，它没有内置的布局或样式，因此必须在模板中显式声明所有的 HTML 布局、格式设置和样式标记。此外，与 DataList 控件和 DataGrid 控件不同的是，Repeater 控件没有内置的选择和编辑支持，用户需通过 Repeater 控件的 ItemCommand 事件来处理从模板引发到该控件的控件事件。

在 Repeater 控件中，提供了五种不同类型的模板：AlternatingItemTemplate、FooterTemplate、HeaderTemplate、ItemTemplate 和 SeparatorTemplate。其中，AlternatingItemTemplate 模板用于控制交替项内容和布局；FooterTemplate 模板用于控制列表注脚的内容和布局；HeaderTemplate 模板用于控制列表标头的内容和布局；ItemTemplate 模板用于控制列表中各项目的内容和布局；SeparatorTemplate 模板则用于控制呈现在各项之间的内容和布局。在以上模板中，ItemTemplate 模板是必选模板，而其他模板则可有可无。

Repeater 控件的常用属性如下。

- AlternatingItemTemplate：用于获取或设置 Repeater 控件中的交替项的模板。

- DataMember：用于获取或设置 DataSource 中要绑定到 Repeater 控件的特定表。
- DataSource：用于获取或设置为填充 Repeater 控件提供数据的数据源。
- FooterTemplate：用于获取或设置 Repeater 控件中的注脚部分的模板。
- HeaderTemplate：用于获取或设置 Repeater 控件中的标头部分的模板。
- Items：用于获取 Repeater 控件中的所有项的集合。
- ItemTemplate：用于获取或设置 Repeater 控件中的各项目的模板。

6.6　典型实例：使用 DataList 和 Repeater 控件绑定数据源

在 6.5 节中，大家对 DataList 控件和 Repeater 控件已经有了一个初步的了解。在本节中，将通过一个具体的实例来介绍 DataList 控件和 Repeater 控件的实际应用。

在此实例中，将分别创建一个 DataList 控件和 Repeater 控件，并将它们绑定至同一数据源上，该数据源与本章前面示例的数据源完全一样。

页面代码如下：

【DataList_Repeater.aspx】

```
<%@ Page Language="VB" ContentType="text/html" ResponseEncoding="gb2312" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<%@ Import NameSpace="System.Data" %>
<%@ Import NameSpace="System.Data.OleDb" %>
<Script runat="server">
  Private Sub Page_Load(Byval Sender As Object,Byval E As EventArgs)
    If Not IsPostBack Then
      BindData()  '调用自定义过程 BindData，绑定数据
    End If
  End Sub
  Sub BindData()
    Dim Cnn As OleDbConnection
    Dim Cmd As OleDbCommand
    Dim dtCmd As OleDbDataAdapter
    Dim DS As DataSet
    '创建数据库连接
    Cnn=new OleDbConnection("Provider=Microsoft.Jet.OleDb.4.0;Data
Source=" & Server.MapPath("Friend.mdb"))
    Cnn.open()
    '查询数据
    Cmd=New OleDbCommand("Select * from MyFriends order by id",Cnn)
    dtCmd=New OleDbDataAdapter(Cmd)
    DS=New DataSet
    dtCmd.Fill(DS,"friend")
    '将数据绑定至 DataList 控件中
    DataList1.DataSource=DS.Tables("friend")
    DataList1.DataBind()
```

```
    '将数据绑定至 Repeater 控件中
    Repeater1.DataSource=DS.Tables("friend")
    Repeater1.DataBind()
    Cnn.Close()
  End Sub
</Script>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<title>使用 DataList 控件和 Repeater 控件</title>
</head>
<body>
<form id="Form1" runat="server">
<p align="center"><font color="#0000FF"><b>使用 DataList 控件</b></font></p>
<p align="center">
<asp:DataList id="DataList1" runat="server" width="480"
BorderColor="black" CellPadding="5" RepeatLayOut=Table RepeatColumns="3"
GridLines="Both" Font-Size="11pt">
  <HeaderStyle BackColor="#aaaadd"></HeaderStyle>
  <HeaderTemplate><center><b>我的朋友</b></center></HeaderTemplate>
  <ItemTemplate>
    <%# Container.DataItem("Name") %>
  </ItemTemplate>
</asp:DataList></p><br />
<p align="center"><font color="#0000FF"><b>使用 Repeater 控件</b></font></p>
<p align="center">
<asp:Repeater id="Repeater1" Runat="Server">
  <HeaderTemplate>
    <Table width="80%" border="0" style="font-size:14px">
    <tr height="30">
      <th>姓名</th><th>性别</th><th>家庭住址</th><th>联系电话</th><th>邮政编码
</th><th>工作单位</th>
    </tr>
  </HeaderTemplate>
  <ItemTemplate>
    <tr height="30">
      <td align="center"><%# Container.DataItem("Name") %></td>
      <td align="center"><%# Container.DataItem("Sex") %></td>
      <td><%# Container.DataItem("HomeAddr") %></td>
      <td><%# Container.DataItem("TelPhone") %></td>
      <td align="center"><%# Container.DataItem("Zipcode") %></td>
      <td><%# Container.DataItem("works") %></td>
    </tr>
  </ItemTemplate>
  <AlternatingItemTemplate>
    <tr bgcolor="#CCFFFF">
      <td align="center"><%# Container.DataItem("Name") %></td>
      <td align="center"><%# Container.DataItem("Sex") %></td>
      <td><%# Container.DataItem("HomeAddr") %></td>
      <td><%# Container.DataItem("TelPhone") %></td>
      <td align="center"><%# Container.DataItem("Zipcode") %></td>
      <td><%# Container.DataItem("works") %></td>
```

```
      </tr>
    </AlternatingItemTemplate>
    <FooterTemplate></table></FooterTemplate>
  </asp:Repeater></p>
  </form>
  </body>
  </html>
```

在代码中，表达式<%# Container.DataItem("字段名")%>是 DataList 控件和 Repeater 控件中的一个非常关键的要素，它用于获取绑定的数据源中指定字段的值。该表达式也可使用表达式<%# DataBinder.Eval(Container.DataItem,"字段名") %>来替换，两者代表同样的含义。

从以上代码可以看出，DataList 控件比 Repeater 控件更简单。但是，Repeater 控件却具有更为强大的灵活性和可扩展性。同时，Repeater 控件对设计人员在 HTML 语言方面的掌握程度也要求更高。

页面的预览效果如图 6.6 所示。

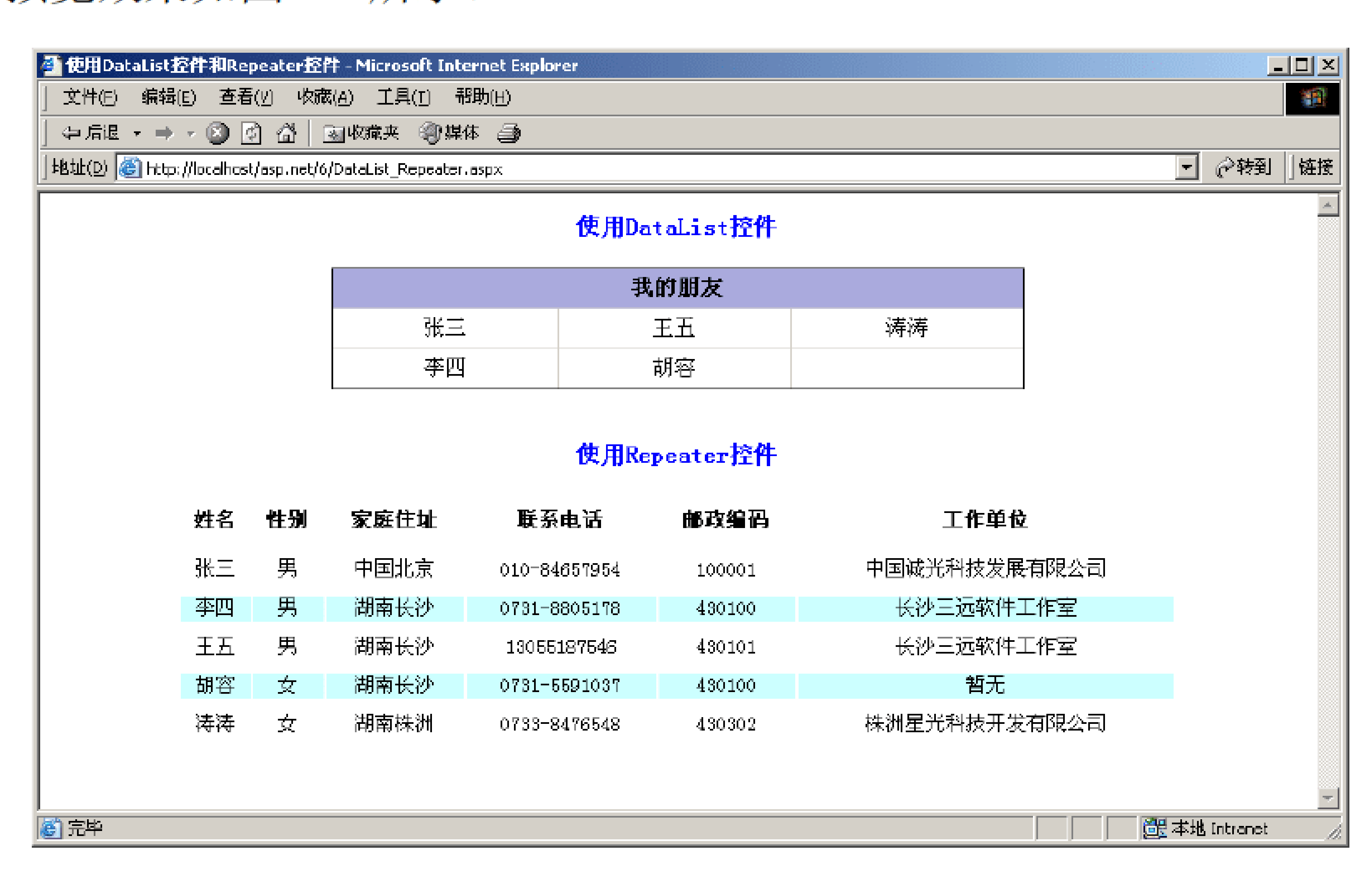


图 6.6　页面预览效果

6.7　习　　题

(1)　练习 ADO.NET 的几种常用对象的具体使用。

(2)　练习如何直接通过代码来创建 DataSet 对象。

(3)　扩展 6.4.2 小节中的 DataGrid 应用实例，添加 DataGrid 的排序功能。

(4)　扩展 6.6 节典型实例，在 DataList 控件和 Repeater 控件中添加分页功能。

第 7 章　留言簿制作

通过前面章节的学习，相信读者对 Dreamweaver 8 和 ASP.NET 以及 Access 数据库都有了一个基本的了解。在后续的章节中，将为大家介绍如何通过 Dreamweaver 8 和 ASP.NET 来开发 Web 应用程序，包括留言簿、网络相册、网站会员系统、同学录、新闻发布系统以及目前流行的博客系统等。

留言簿一直是网站的重要组成部分，它为浏览者与网站管理员提供了一个直接进行交流的通道。本章，将介绍如何通过 Dreamweaver 和 ASP.NET 来开发留言簿系统。

7.1　系 统 分 析

7.1.1　系统功能

留言簿作为浏览者与网络管理员交互的桥梁，浏览者可以发表自己的浏览信息，查看管理员的回复以及他人的留言和相关回复；而管理员可以对留言进行管理，包括回复留言、删除留言等。

由于留言簿是面向所有浏览者，为了保证留言信息的个人隐私，因此浏览者在发表留言时可使用“悄悄话”功能，即该留言仅管理员和自己可以看到，而其他浏览者无法看到。此外，对于某些留言，如果涉及公共道德或他人隐私，则管理员也可对其进行屏蔽。屏蔽与删除的区别在于，前者并不删除留言信息，而仅仅是屏蔽其留言内容，使他人无法看到；后者则是将留言信息彻底删除，不留痕迹。

这里，将留言簿系统的功能划分为前台页面功能和后台页面功能。前台页面功能是指一般浏览者可以直接操作的功能，而后台页面功能是面向管理员的，只有进行登录并确认身份后才可进行操作。事实上，对于网站中的大多数的 Web 应用程序，均有前台和后台之分。通过后台，管理员可以对一般用户在前台所操作的信息进行管理。

1. 前台页面功能

前台页面主要针对一般浏览者，其功能包括发布留言和浏览留言。

1)　发布留言

发布新的留言，其内容包括留言标题、留言人、性别、个人头像、邮箱地址、联系 QQ、个人主页、留言表情和留言内容，以及悄悄话等。

2)　浏览留言

浏览所有的留言以及管理员的相关回复内容。在此功能中，需注意留言是否为悄悄话或是否已被管理员屏蔽。如果留言被设置为悄悄话，则当一般用户浏览时，应向其显示“此留言为悄悄话”，而对管理员则显示真实的留言内容。如果留言已被管理员屏蔽，则应向

浏览者显示“此留言已被管理员屏蔽”。

2. 后台页面功能

后台页面主要针对管理员，其功能包括管理员登录、回复留言、删除留言和屏蔽留言。

1)　管理员登录

提供管理员的登录界面，通过在页面上所输入的用户名和用户密码来确认管理员的真实身份。一旦核实正确，则进行管理页面；否则，提示相应的错误信息。

2)　回复留言

对指定的留言进行回复，其信息包括回复内容和回复时间。如果过去已对指定的留言进行过回复，则可对回复的内容进行修改，并覆盖过去的回复信息。

3)　删除留言

对于指定的留言信息，将其从数据库中彻底删除。

4)　屏蔽留言

对于不健康或涉及他人隐私的留言信息，将其状态设置为屏蔽，以确保该留言信息不会在留言浏览页面中显示其真实的留言信息。

7.1.2　数据库的建立

本系统采用 Microsoft Access 2000 作为数据库，数据库名为 Data.Mdb，在该数据库中添加了 UserInfo(管理员信息表)、LYInfo(留言信息表)、FaceInfo(可选头像信息表)和 BiaoQing(可选表情信息表)4 个数据表。

1. 管理员信息表(UserInfo)

UserInfo 数据表主要用于存储管理员的基本信息，其表结构见表 7.1。

表 7.1　UserInfo 数据表

字 段 名	数据类型	字段描述	字段大小	默 认 值	主　键
ID	自动编号	管理员的编号	长整型		P
UserName	文本	登录名称	50		
UserPwd	文本	登录密码	50		

说明：在主键一栏中，P(Primary)表示该字段为数据表的主键。

2. 留言信息表(LYInfo)

LYInfo 数据表主要用于存储留言内容，这也是本系统中最关键和最重要的数据表，其表结构见表 7.2。

表 7.2　LYInfo 数据表

字 段 名	数据类型	字段描述	字段大小	默 认 值	主　键
ID	自动编号	留言编号	长整型		P
Name	文本	留言人姓名	50		

续表

字 段 名	数据类型	字段描述	字段大小	默 认 值	主　键
Sex	文本	留言人的性别	4		
QQ	文本	留言人的联系 QQ	50		
E-mail	文本	留言人的邮箱地址	100		
Homepage	文本	留言人的个人主页	100		
Title	文本	留言主题	100		
Content	备注	留言内容			
Ddate	日期/时间	留言时间		Now()	
Reply	备注	回复内容			
ReplyDate	日期/时间	回复时间			
FaceID	数字	留言人头像的 ID，对应于 FaceInfo 数据表中的 ID 字段			
BiaoQingID	数字	留言表情的 ID，对应于 BiaoQing 数据表中的 ID 字段			
Tag_QQH	文本	是否为悄悄话；其值为 1 表示此留言为悄悄话，为 0 表示非悄悄话	1	0	
Tag_Hide	文本	是否屏蔽；其值为 1 表示此留言已被管理员屏蔽，为 0 表示正常	1	0	

说明： 字段 Ddate 的默认值为系统函数 Now()，该函数用于系统的当前时间。即在添加新记录时，如果没有为字段 Ddate 指定数据，则获取系统的当前时间作为字段的值。

3. 可选头像信息表(FaceInfo)

FaceInfo 数据表主要用于存储可选头像的相关信息，其表结构见表 7.3。

表 7.3　FaceInfo 数据表

字 段 名	数据类型	字段描述	字段大小	默 认 值	主　键
ID	自动编号	头像编号	长整型		P
Name	文本	头像名称	50		
PicAddr	文本	头像所对应的图片地址	50		

4. 表情信息表(BiaoQing)

BiaoQing 数据表主要用于存储留言所用表情图标的相关信息，其表结构见表 7.4。

事实上，可选头像信息表(FaceInfo)和表情信息表(BiaoQing)均不是系统必要的数据表。在添加留言信息时，用户可指定固定的头像图片和表情图片，但那样势必给系统的后期维护带来不便。一旦需要添加新的头像和表情，均需修改系统相关的文件源代码，这是非常

麻烦的。因此，为了保持系统良好的可扩展性，这里将头像图片和表情图片的相关信息通过数据表来存储。用户只需在后台功能页面中添加头像图片和表情图片，前台的添加留言页面即可自动更新。

表 7.4　BiaoQing 数据表

字 段 名	数据类型	字段描述	字段大小	默 认 值	主　键
ID	自动编号	留言表情编号	长整型		P
Addr	文本	留言表情所对应的图片地址	50		

7.1.3　站点设置

在正式介绍系统功能的实现之前，必须做一些准备工作，这就是对站点的设置，这一步也是通过 Dreamweaver 8 开发基于 ASP.NET 的 Web 应用程序的前提。

在第 2 章中，对站点的设置进行了详细的讲解，这里就不再赘述。本例中，站点设置的本地信息如图 7.1 所示。

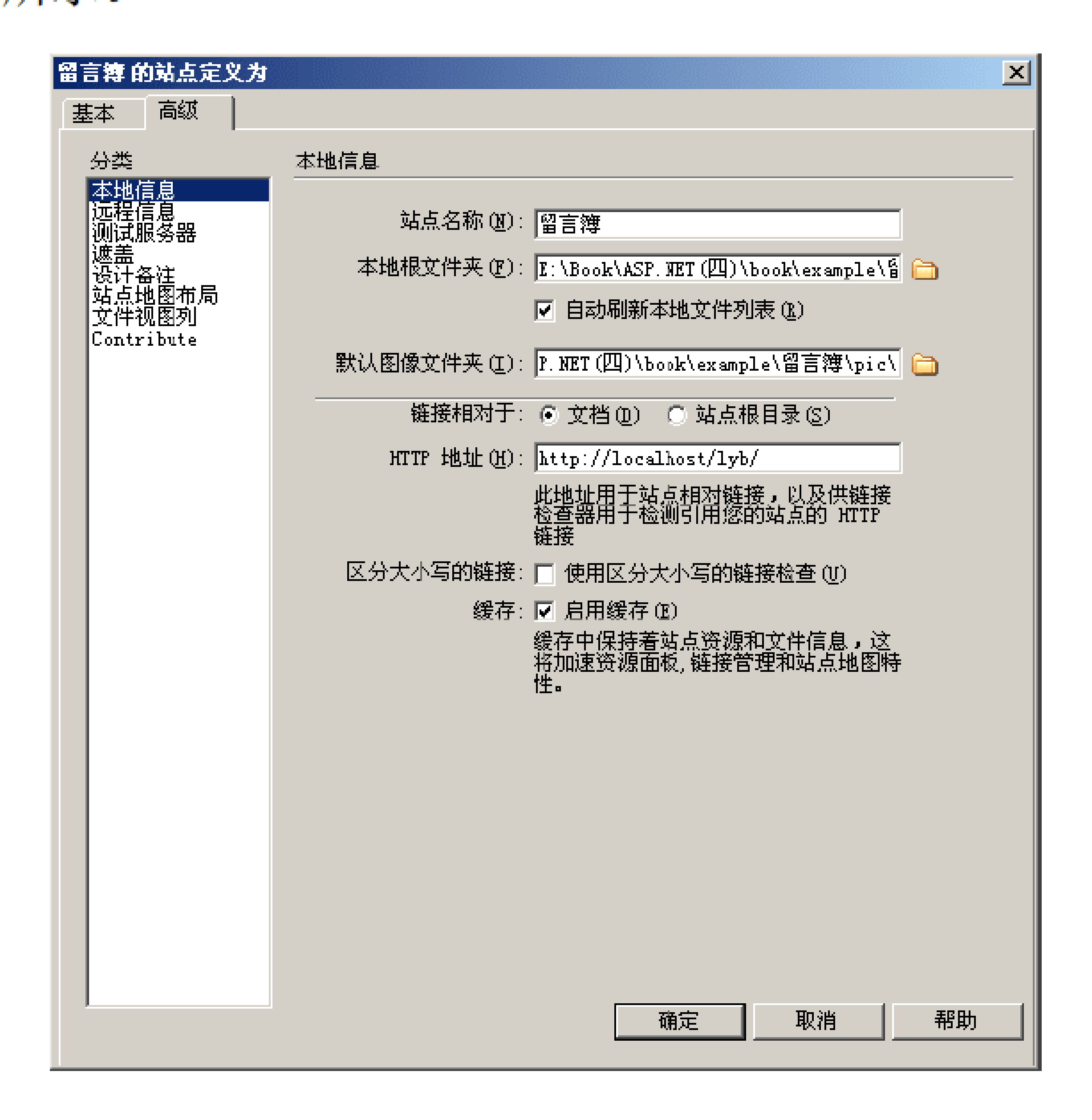


图 7.1　本地信息

其中，【站点名称】设置为“留言簿”，【本地根文件夹】设置为留言簿系统所在的磁盘目录，【默认图像文件夹】则设置为留言簿系统下的 pic 文件夹。此外，【HTTP 地址】设置为 http://localhost/lyb，这意味着在此之前须将留言簿系统设置为 Web 共享，其共享名为 lyb，这样便可通过该 HTTP 地址来访问留言簿。

站点设置中，远程信息设置如图 7.2 所示。

由于对留言簿系统的设计、测试和运行均是在本机上操作的，因此这里将访问方式设置为【本地/网络】，而远端文件夹则与本地根文件夹设置为相同的目录。

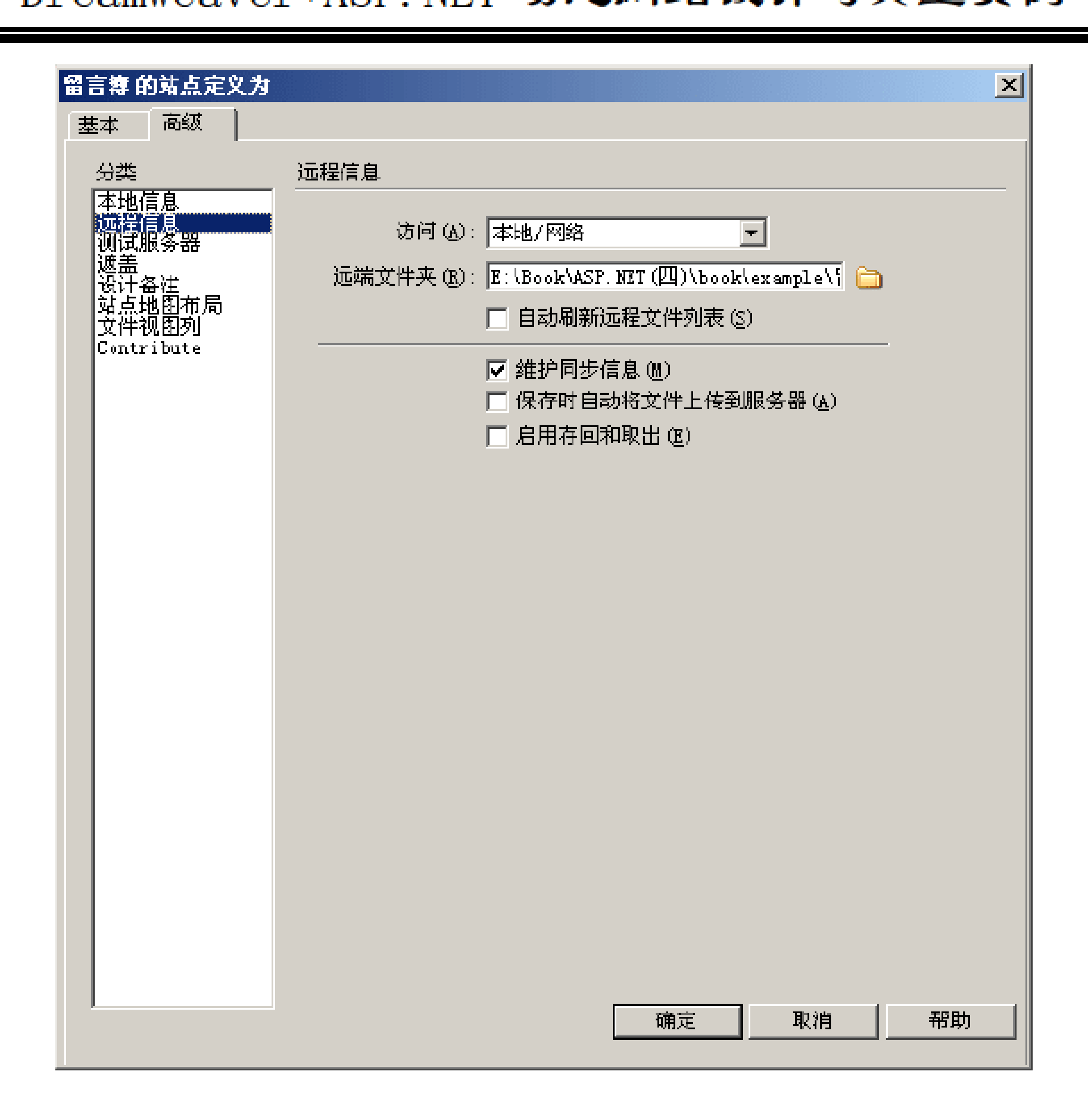


图 7.2　远程信息

站点设置中，测试服务器设置如图 7.3 所示。

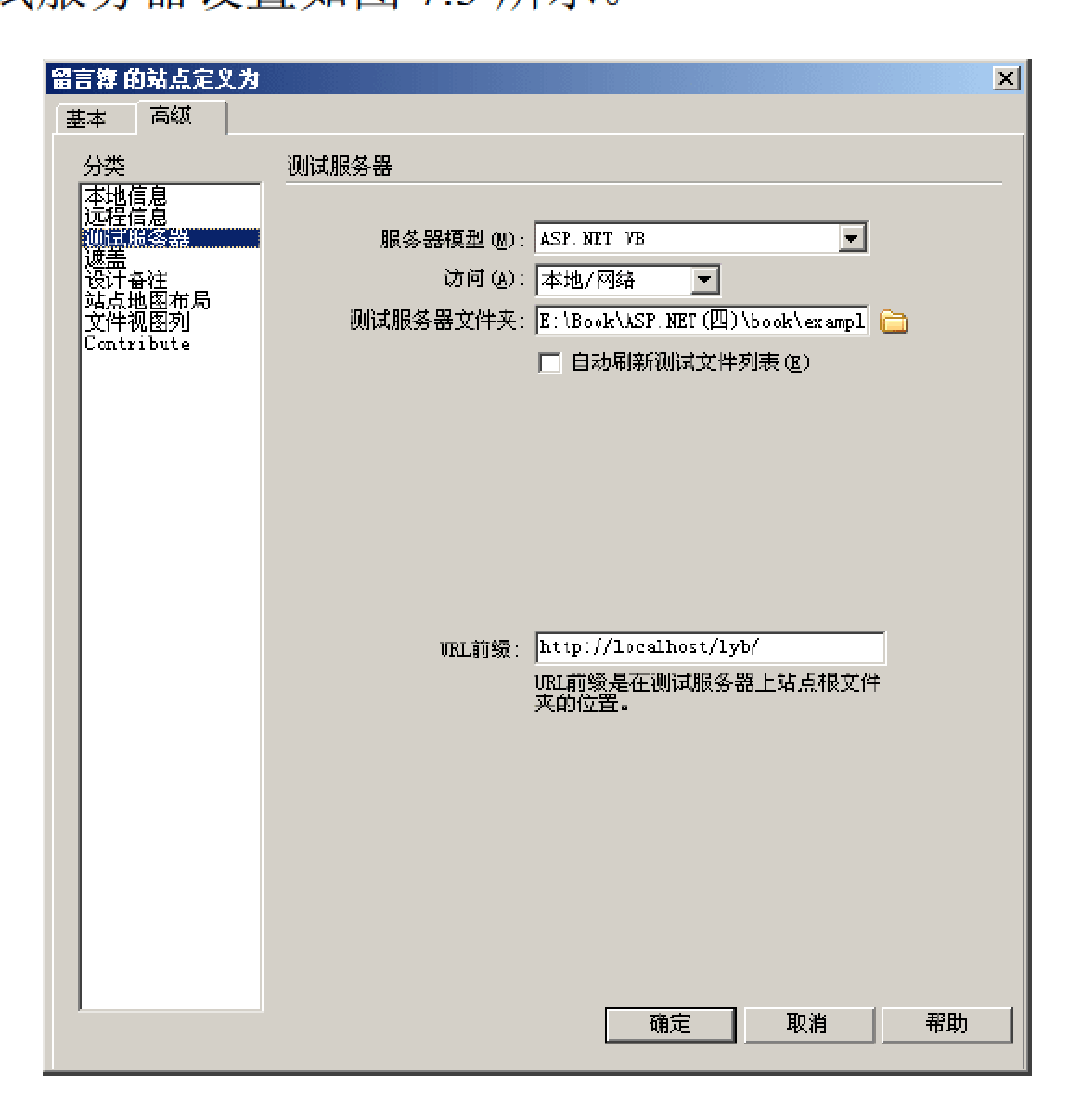


图 7.3　测试服务器

请注意，如果需要 Dreamweaver 8 来调用数据库中的动态数据，则测试服务器是必须设置的，否则无法对程序进行部署。在这里，同样将测试服务器文件夹设置为与本地文件夹相同的目录。

站点设置完成后，还有一步极为重要的操作，那就是站点的部署。对于大多数的初学者，往往容易将此步操作忽略，结果导致程序无法正常运行。

所谓站点的部署，实质上是部署 DreamweaverCtrls.dll 控件。该控件属于复合控件，是 Dreamweaver 8 在 ASP.NET 页面中用于处理数据的关键控件。事实上，在 Dreamweaver 8

中，所有的 ASP.NET 核心程序都已集成在 DreamweaverCtrls.dll 控件中。

如果没有部署 DreamweaverCtrls.dll 控件至应用程序根目录下的 Bin 目录中(即使在页面中插入了数据集和数据网络，并且在数据集对话框中测试时能读出数据)，则在页面实际浏览时将始终会看到“系统找不到指定的文件”之类的错误信息，如图 7.4 所示。

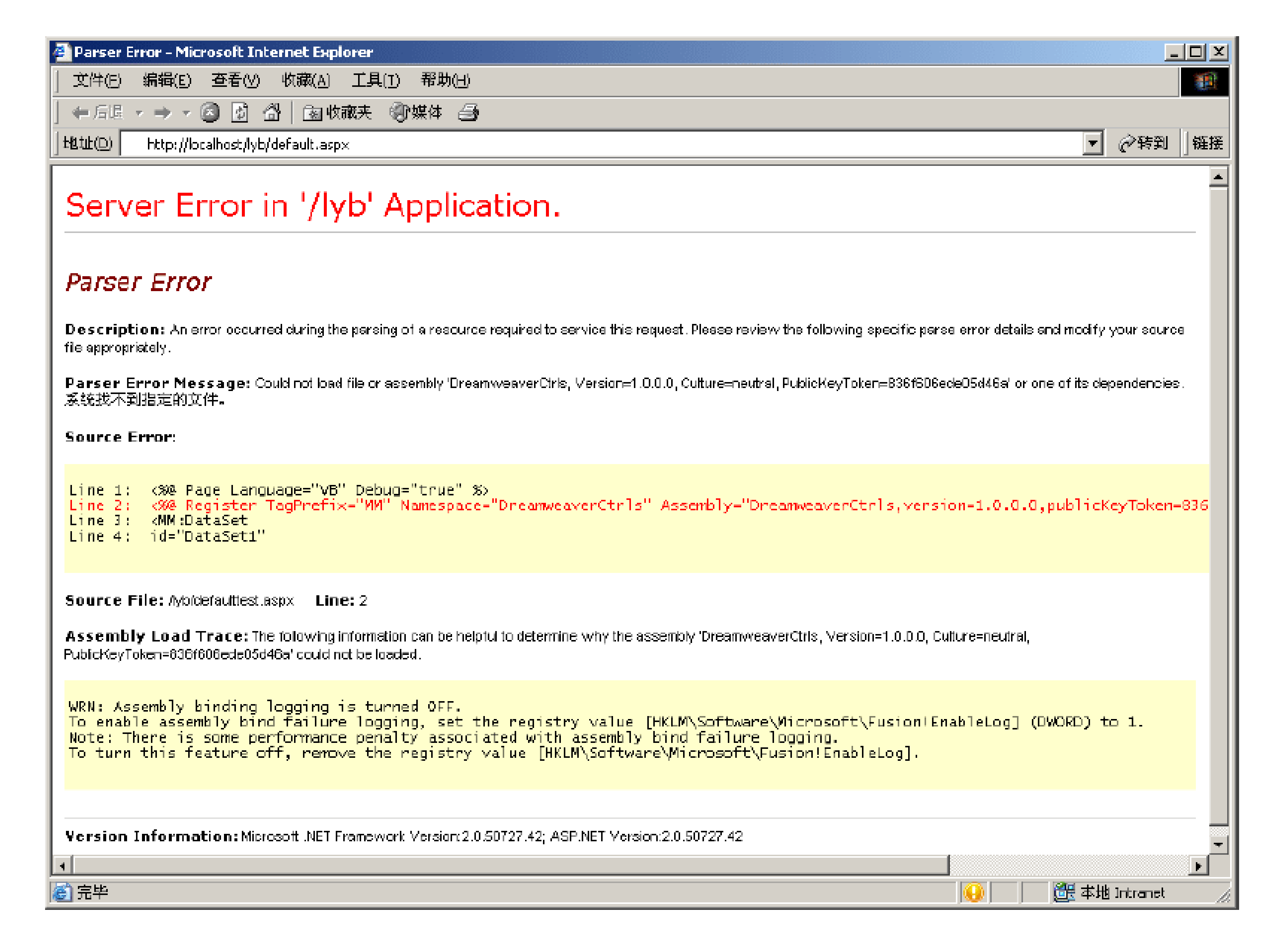


图 7.4　错误信息

下面，就来部署 DreamweaverCtrls.dll 控件。

(1)　选择【文件】|【新建】命令，将弹出【新建文档】对话框，如图 7.5 所示。

(2)　在【类别】列表框中选择【动态页】，在【动态页】列表框中选择 ASP.NET VB，然后单击【创建】按钮，这样即可创建一个基于 VB.NET 语言的 ASP.NET 页面。此时，在页面的【设计】视图中将是一片空白，因为还没有添加任何元素。

(3)　打开【应用程序】面板组，切换至【绑定】面板，如图 7.6 所示。

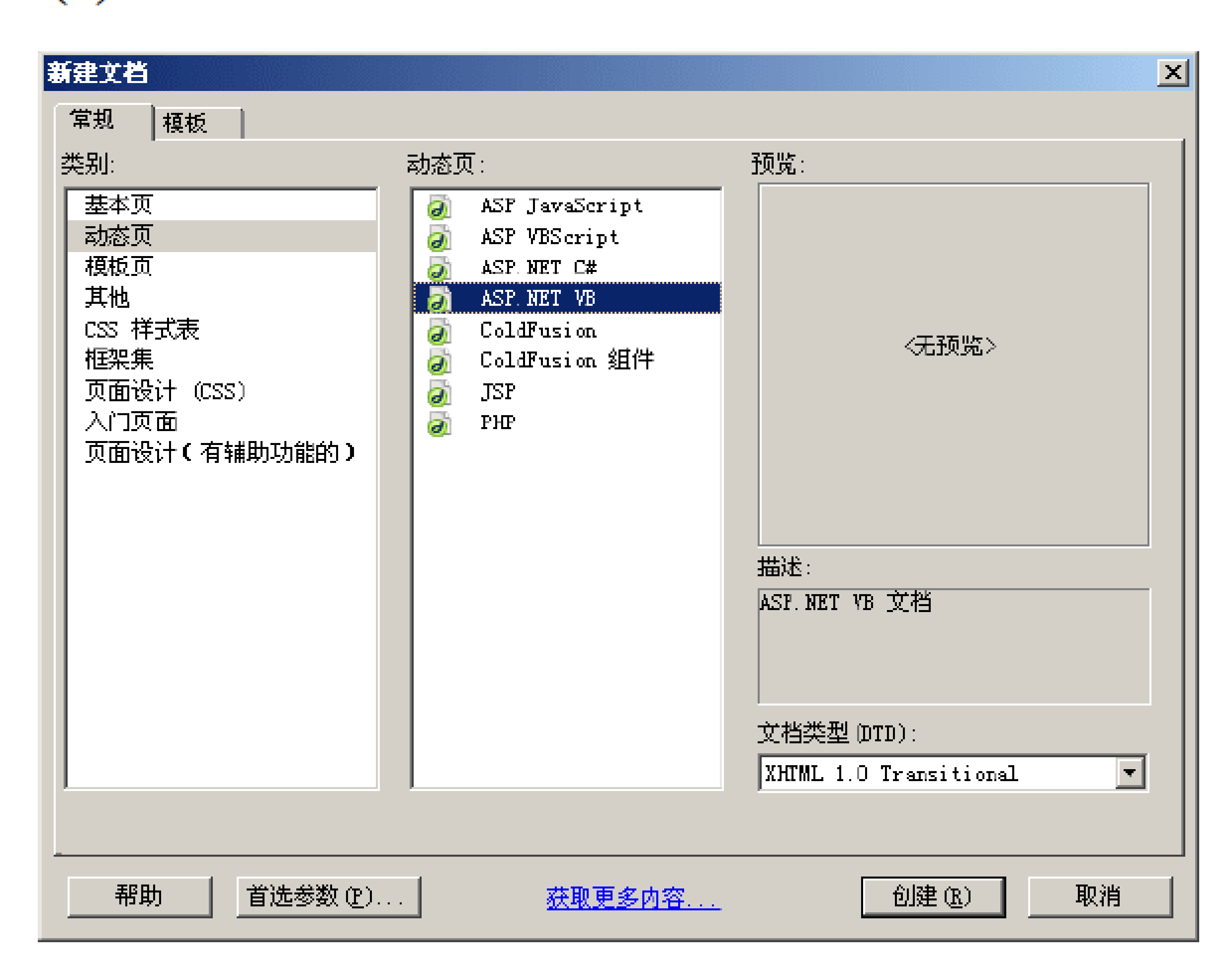


图 7.5　【新建文档】对话框

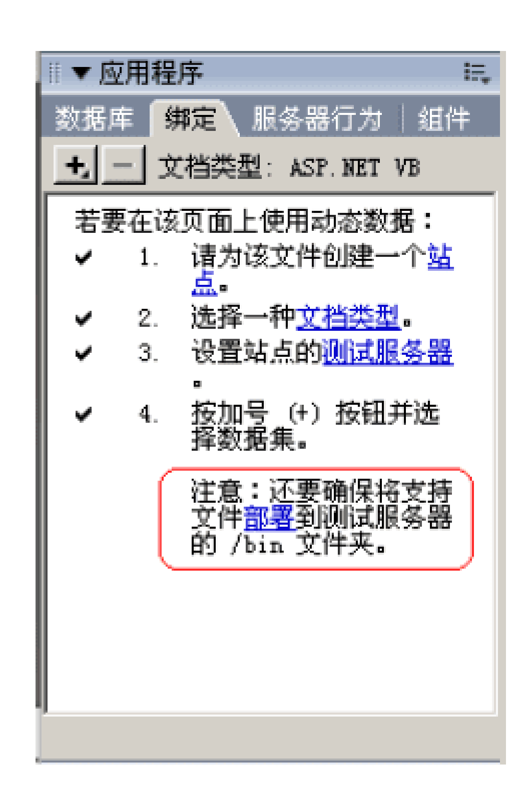


图 7.6　【绑定】面板

(4) 单击【部署】链接，将弹出【将支持文件部署到测试服务器】对话框，如图 7.7 所示。

(5) 单击【文件夹】按钮，在弹出的对话框中，选择程序目录下的 Bin 文件夹，然后单击【部署】按钮，此时系统将弹出一个对话框提示部署成功，如图 7.8 所示。

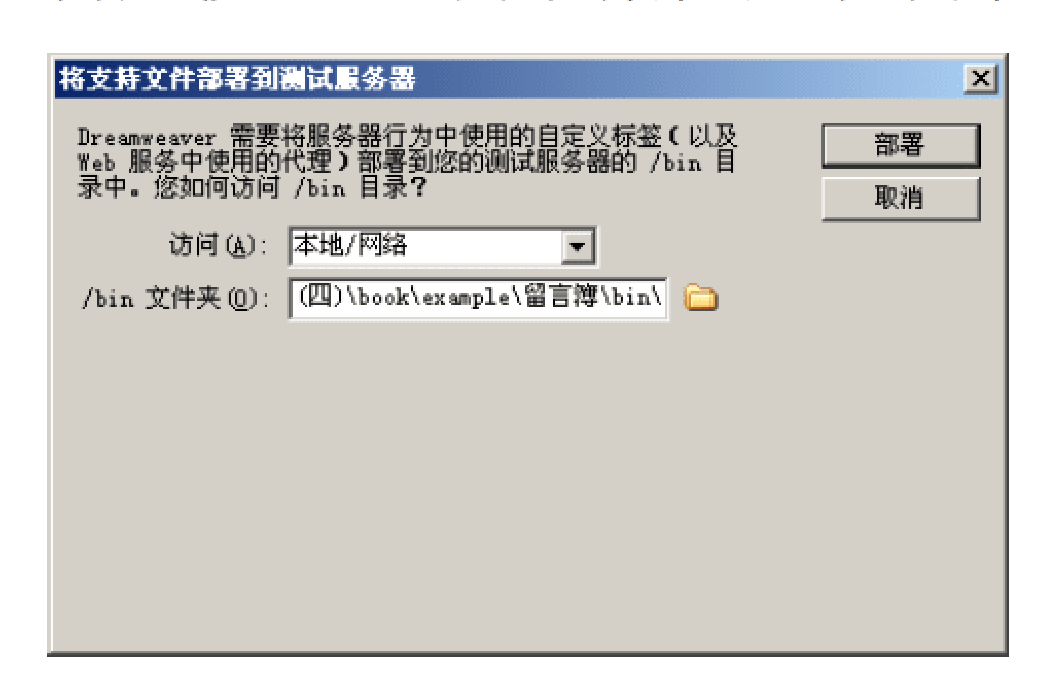


图 7.7　【将支持文件部署到测试服务器】对话框

图 7.8　部署成功

打开留言簿根目录下的 Bin 目录，可以发现其中多了一个 DreamweaverCtrls.dll 文件。至此，站点的部署工作成功完成。

7.2　留言簿的前台页面制作

留言簿的前台页面包括浏览留言页面和添加留言页面，其对应的文件分别为 Default.aspx 和 Ly_Add.aspx。

7.2.1　浏览留言页面

1.　页面布局设计

下面，来看一下浏览留言页面的制作。

(1) 选择【文件】|【新建】命令，将弹出【新建文档】对话框，如图 7.9 所示。

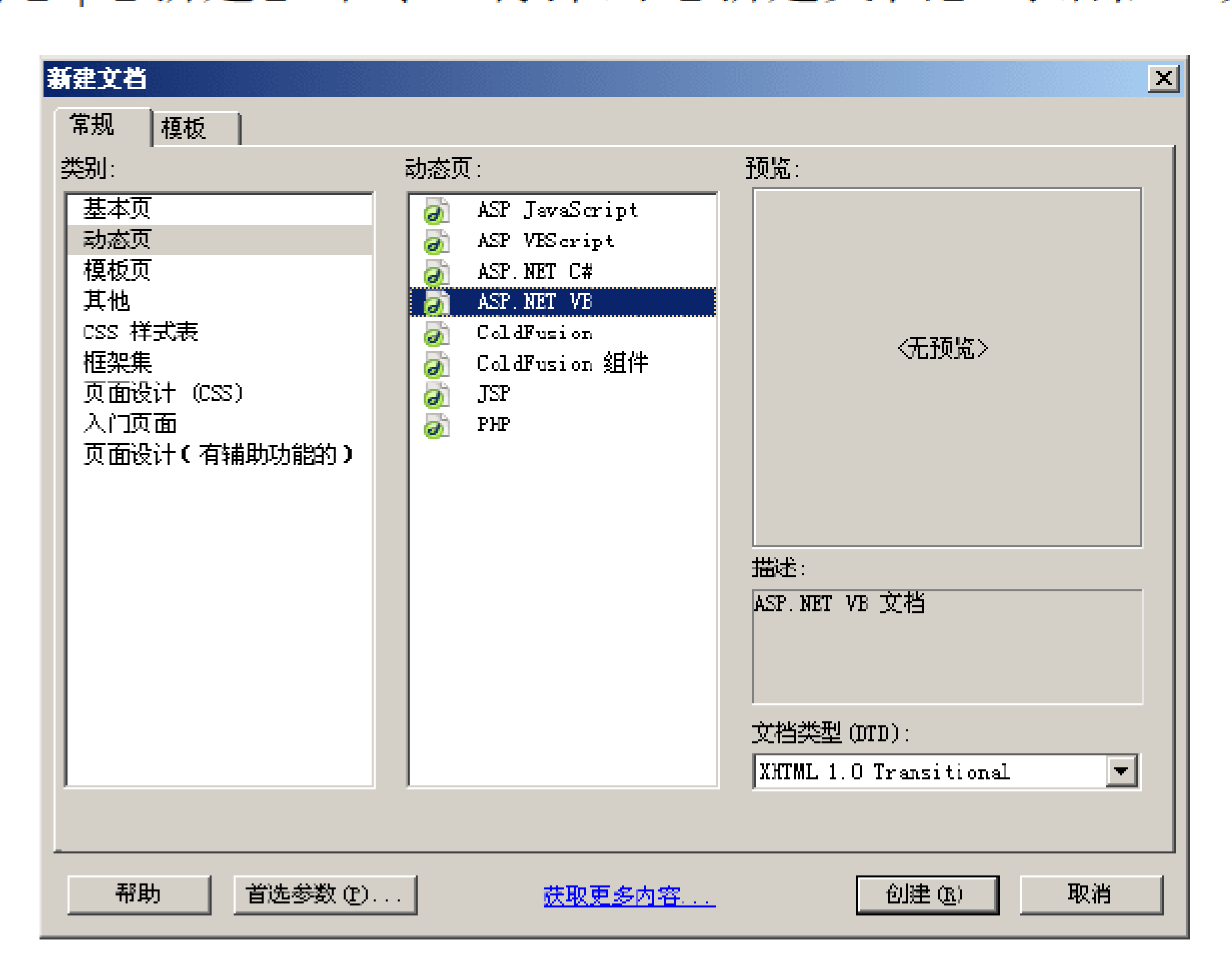


图 7.9　【新建文档】对话框

(2) 在【类别】列表框中选择【动态页】，在【动态页】列表框中选择 ASP.NET VB，然后单击【创建】按钮，这样即可创建一个基于 VB.NET 语言的 ASP.NET 页面。此时，在页面的【设计】视图中将是一片空白，因为我们还没有添加任何元素。

(3) 在【设计】视图中右击，从弹出的快捷菜单中选择【页面属性】命令，在弹出的【页面属性】对话框中，将背景颜色值设置为“#9898BA”，上边距设置为 10 像素，如图 7.10 所示。

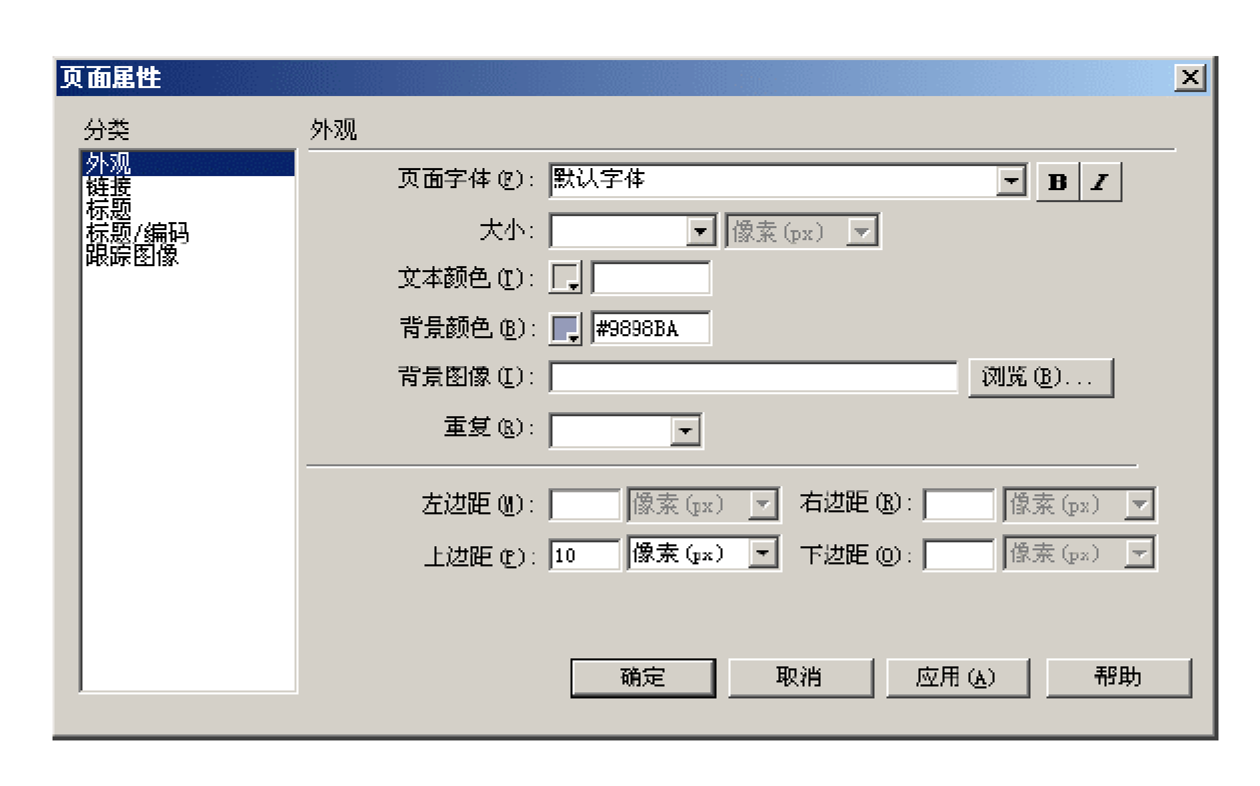


图 7.10　【页面属性】对话框

(4) 单击【确定】按钮，完成页面的设置。在页面的【属性】面板中，打开【样式】下拉列表框，选择【附加样式表】选项，将弹出如图 7.11 所示的对话框。

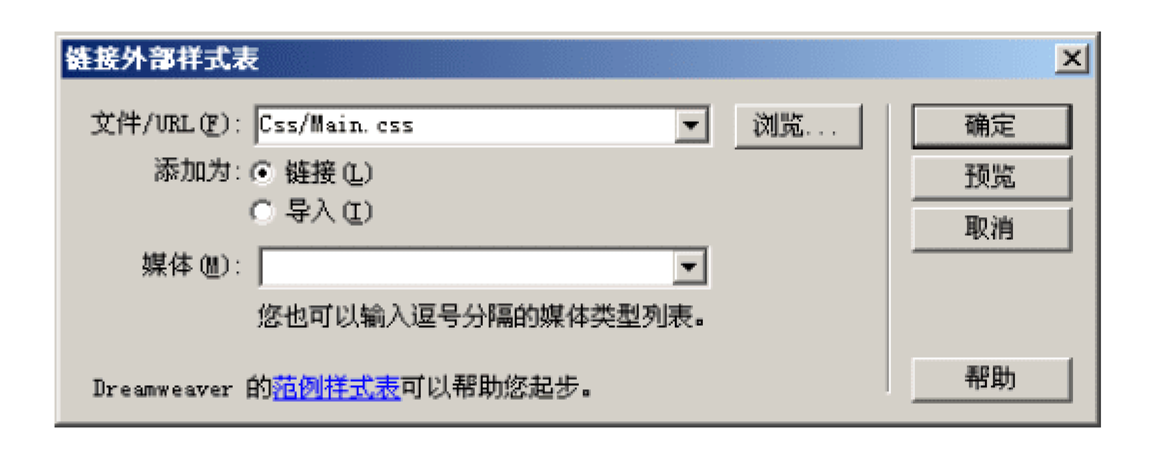


图 7.11　【链接外部样式表】对话框

(5) 单击【浏览】按钮，选择 CSS 文件夹下的 Main.css 文件，再单击【确定】按钮，将该文件插入页面中。此时，在【样式】下拉列表中，将会显示出该文件中所定义的各种样式。

(6) 选择【插入】|【表单】|【表单】命令，在页面中插入一个表单。切换到【代码】视图，为插入的表单添加“Runat="Server"”属性，如下：

```
<form id="form1" name="form1" method="post" action="" Runat="Server">
</form>
```

(7) 将视图切换到【设计】视图，在表单内插入一个表格 Table1，将行数设置为 3，列数设置为 2，表格宽度设置为 750 像素，边框粗细为 0，如图 7.12 所示。

(8) 在表格内右击，从弹出的快捷菜单中选择【表格】|【选择表格】命令，然后在【属性】面板中设置表格的对齐方式为【居中对齐】。

(9) 选择第一行中的两个单元格，单击【属性】面板中的□按钮，合并单元格，并设置其背景为白色(#FFFFFF)，行高为 40，水平对齐方式为【居中对齐】。在单元格中输入

文字“站点留言簿”，设置其字体大小为 16 像素，加粗，字体颜色值为“#9898BA”。

图 7.12　【表格】对话框

(10) 单击表格第 2 行的第 1 个单元格，在【属性】面板中的【样式】下拉列表中，选择样式 td1(该样式用于设置单元格的 Padding_Left 属性为 5 像素)。选择【插入】|【图像】命令，插入用于发表新留言的链接图像，图像文件为 Pic 目录下的 Ly.gif，在【属性】面板中设置其链接至 Ly_add.aspx。

(11) 单击表格第 2 行的第 2 个单元格，在【属性】面板中的【样式】下拉列表中，选择样式 td2 (该样式用于设置单元格的 Padding_Right 属性为 5 像素)。在该单元格中输入文字“管理”，并在【属性】面板中设置其字体为白色(#FFFFFF)，并链接至 Login.aspx。单击该链接，页面将转向管理员的登录页面。

(12) 在表格的下方再新插入一个表格 Table2，该表格主要用于显示留言信息，表格的宽度设为 750 像素，行数为 2，列数为 2，边框粗细为 0。同时在表格的【属性】面板中，设置表格的对齐方式为【居中对齐】，表格的背景颜色值为“#EBEBEB”，填充和间距均为 0。

(13) 选择表格 Table2 的第 1 行，将其高度设置为 25。选择该行的第 1 个单元格，将其样式设置为 td1，背景颜色值设置为“#cccccc”，在其中添加一个 ASP.NET 中的 Image 服务器控件，其操作方法如下。

① 在【插入】工具栏中选择 ASP.NET，如图 7.13 所示。

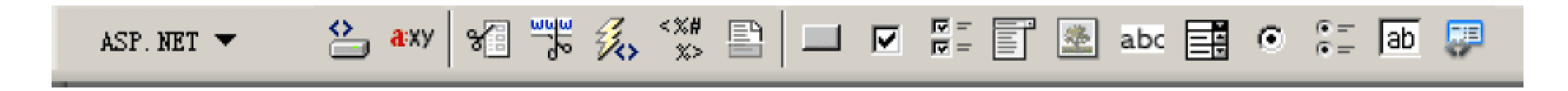


图 7.13　【ASP.NET-插入】工具栏

② 单击【更多标签】按钮，在弹出的【标签选择器】对话框中，选择【ASP.NET 标签】|【Web 服务器控件】分类，在右边的列表框中选择 asp:Image，如图 7.14 所示。

图 7.14 【标签选择器】对话框

③ 单击【插入】按钮，在弹出的【标签编辑器-Image】对话框中设置 ID 为 img_bq，图像 URL 可通过单击【浏览】按钮选择 pic/biaoqing 目录下的任一文件，如图 7.15 所示。

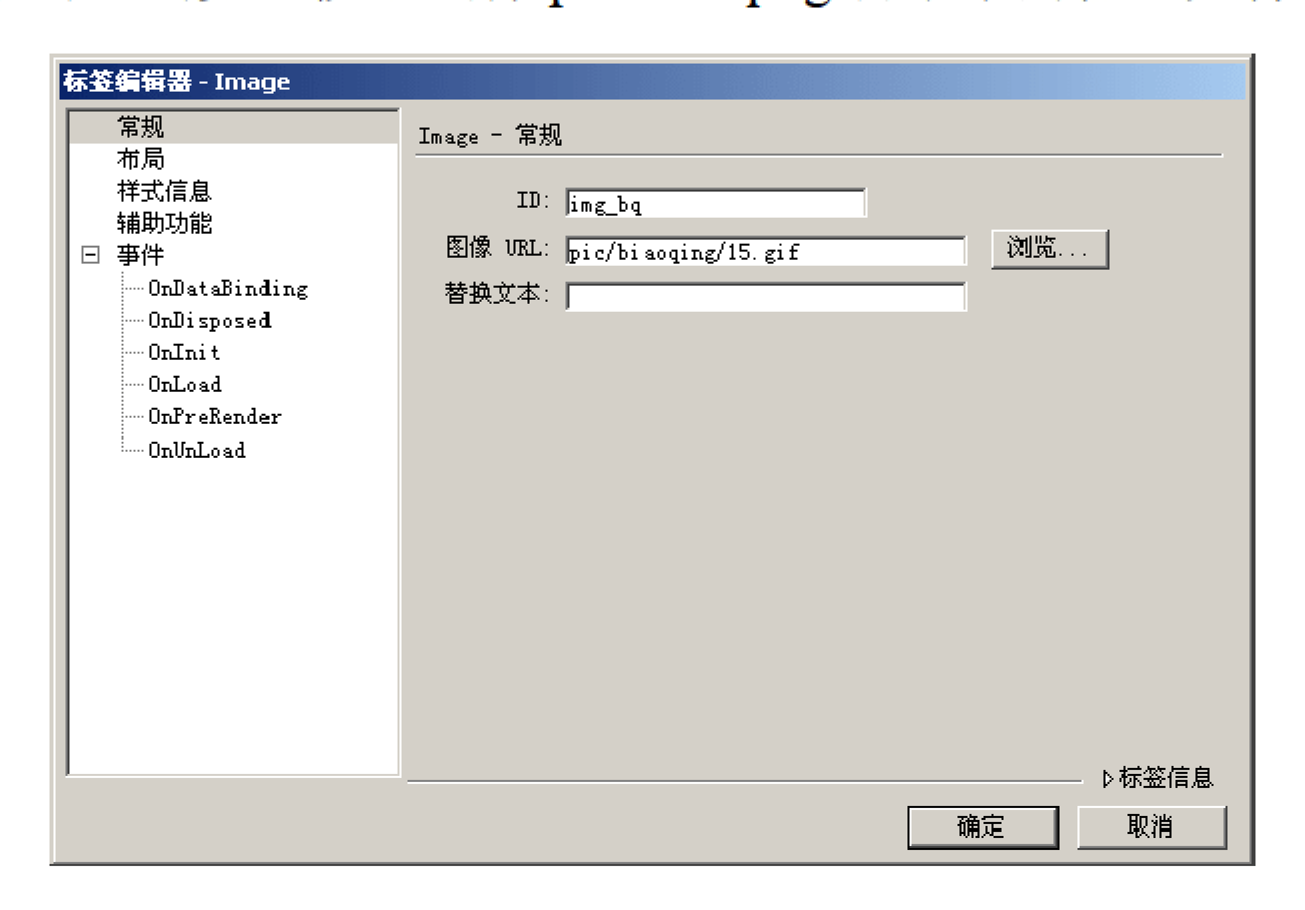


图 7.15 【标签编辑器-Image】对话框

④ 选择【布局】选项，将图像对齐方式设置为【绝对中间】，如图 7.16 所示。

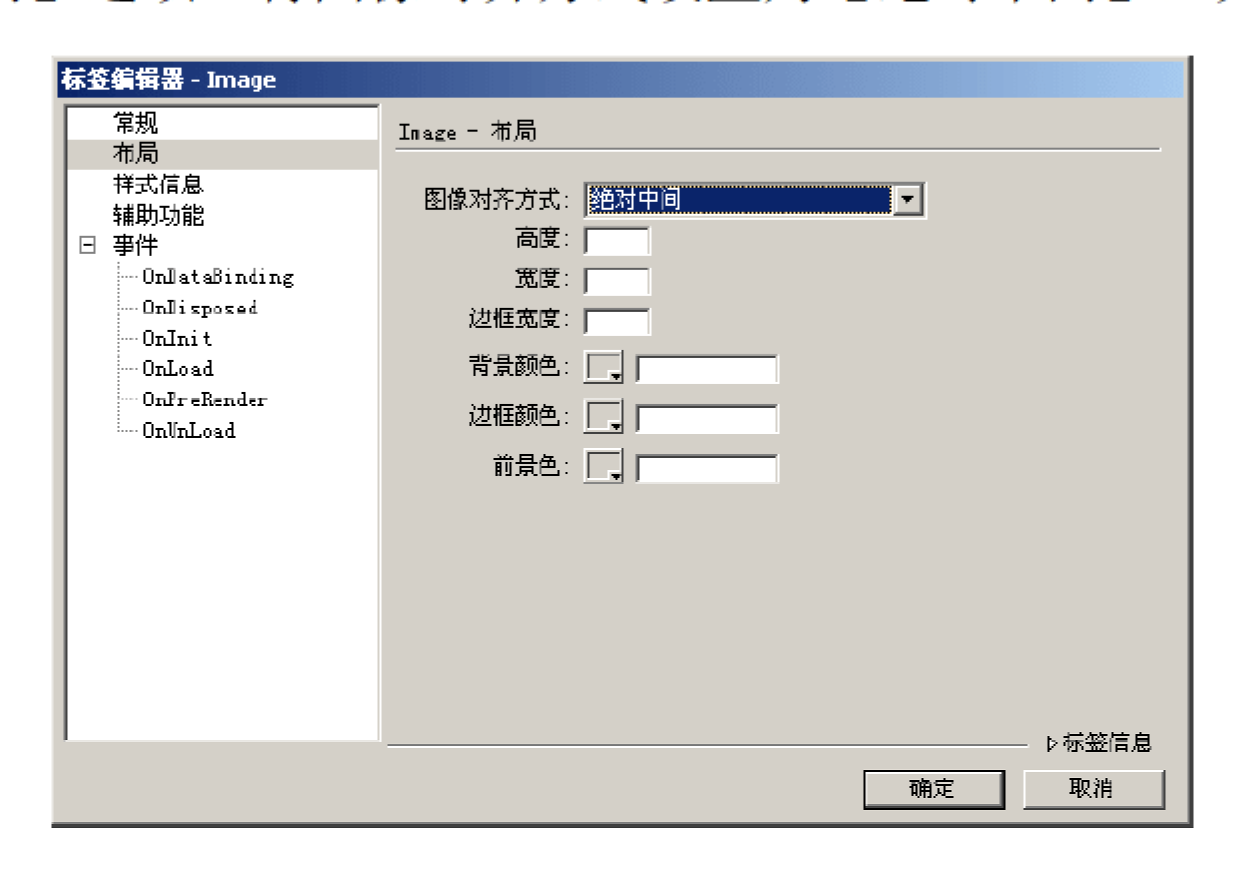


图 7.16 选择【布局】选项

⑤ 单击【确定】按钮，返回【标签选择器】对话框，再单击【关闭】按钮，即可完成插入 Image 服务器控件操作。

这里插入 Image 控件是为了显示用户发表留言时所选择的表情，目前所选择的图片只是临时的，在绑定数据后，将会根据用户发表留言时选择的表情来动态显示。在 Image 控件后，输入文本“标 题：”，并加粗显示。

(14) 选择表格 Table2 的第 1 行的第 2 个单元格，将其样式设置为 td2，背景颜色值设置为“#cccccc”，水平对齐方式选择【右对齐】，在其中插入 3 个图像，其文件分别为 pic 目录下的 Home.gif、QQ.gif、Email.gif。其中，Home.gif 表示留言人的个人主页，QQ.gif 表示留言人的 QQ，Email.gif 表示留言人的电子邮箱地址，其相应的链接将在随后的数据绑定中动态绑定。然后，在【属性】面板中，将 3 个图像的边框属性设置为 0，这样可保证在创建图像链接时将不会出现边框。注意，这里的静态图像与前面的 Image 服务器控件是不同的，前者是静态不变的，而后者是随着数据的不同而显示不同的图像。

(15) 选择表格 Table2 的第 2 行中的两个单元格，单击【属性】面板中的按钮，将其合并，并在其中新插入一个表格 Table3，其宽度设为 750 像素，行数为 2，列数为 3，边框粗细为 0，填充和间距均设置为 0。

(16) 合并表格 Table3 的第一列的两个单元格，设置合并后的单元格的水平对齐方式为【居中对齐】，垂直对齐方式为【居中】，宽度为 120 像素。在此单元格中，插入 ASP.NET 中的 Image 服务器控件，控件 ID 设为 img_face，图片文件任取 pic/ touxiang 目录下的一个文件。此处主要用于显示用户在发表留言时所选择的头像。

(17) 合并表格 Table3 的第 2 列的两个单元格，设置合并后的单元格的背景为白色，宽度为 1 像素。

(18) 选择表格 Table3 的第 3 列的第 2 个单元格，在【属性】面板中设置其水平对齐方式为【右对齐】，设置其样式为 td2，并在其中输入“【留言时间：】”。

至此，一个基本的页面布局已经基本完成，如图 7.17 所示。

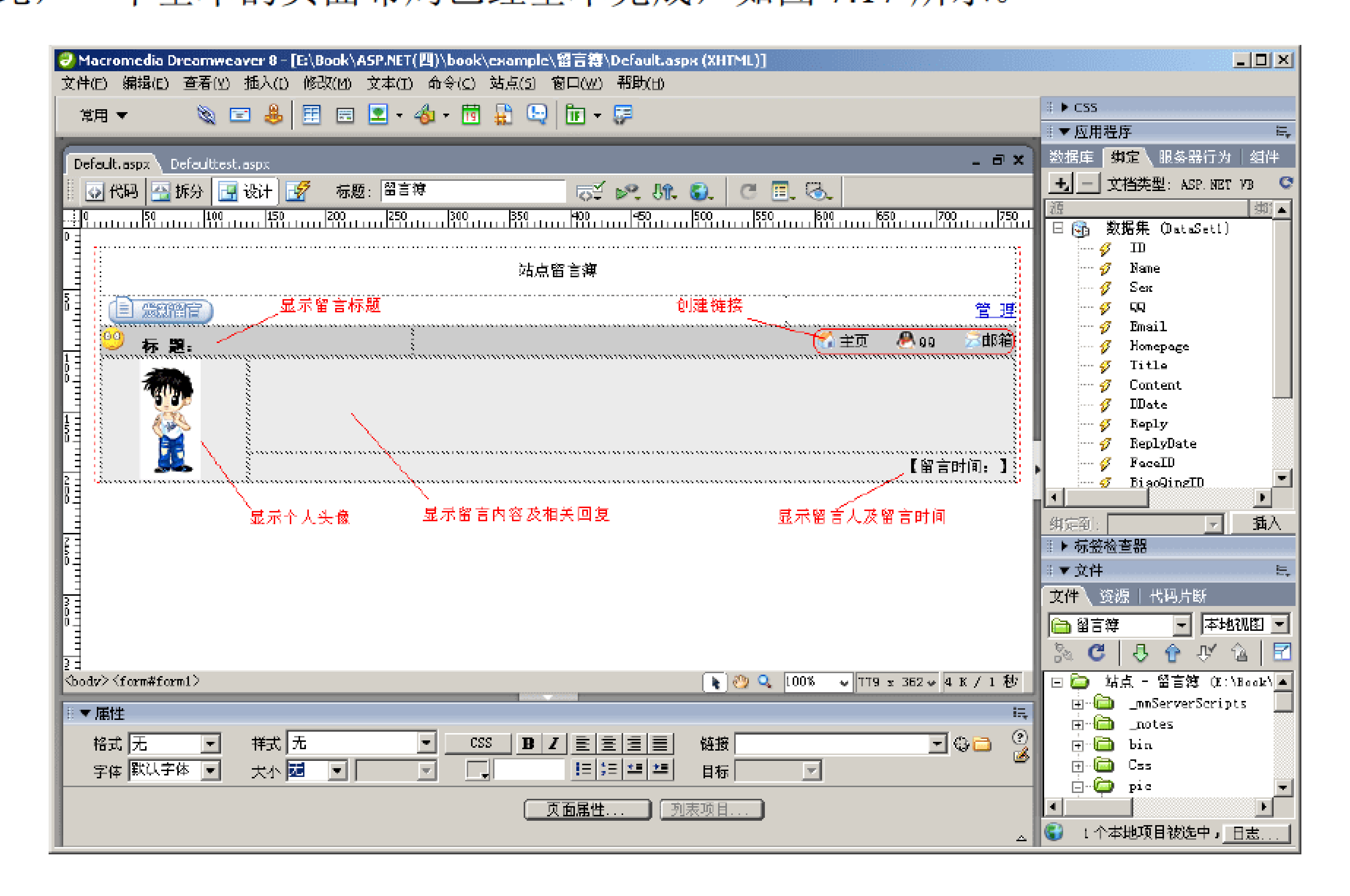


图 7.17　页面的基本布局

在图 7.17 中，用文字标记了各个部分所要显示的信息，而这些信息对应的位置均将插入或绑定动态数据，以便用户浏览时能动态显示。注意，在 Dreamweaver 8 的设计视图中所显示的不一定就是预览的实际结果，因为 CSS 样式在设计视图中并没有体现出来。例如，前面设置了页面的背景颜色值为“#9898BA”，而在设计视图中仍显示为白色。

接下来，看页面数据的绑定，这也是本例中的重中之重。

2. 定义数据库连接

在绑定数据之前，首先需要创建一个数据集。而在创建数据集之前，必须定义一个数据库连接，两者缺一不可。

定义数据库连接的具体操作如下。

(1) 打开【应用程序】面板组，选择【数据库】面板，如图 7.18 所示。

(2) 单击 按钮，系统将弹出菜单，其中包括【OLE DB 连接】和【SQL 服务器】两个命令，如图 7.19 所示。如果在 Web 应用程序中使用的是 Microsoft Access 数据库，则可选择【OLE DB 连接】命令；而对于 SQL Server 数据库，则需选择【SQL 服务器连接】命令。这里选择前者，即 OLE DB 连接。此时，系统将弹出【OLE DB 连接】对话框，如图 7.20 所示。

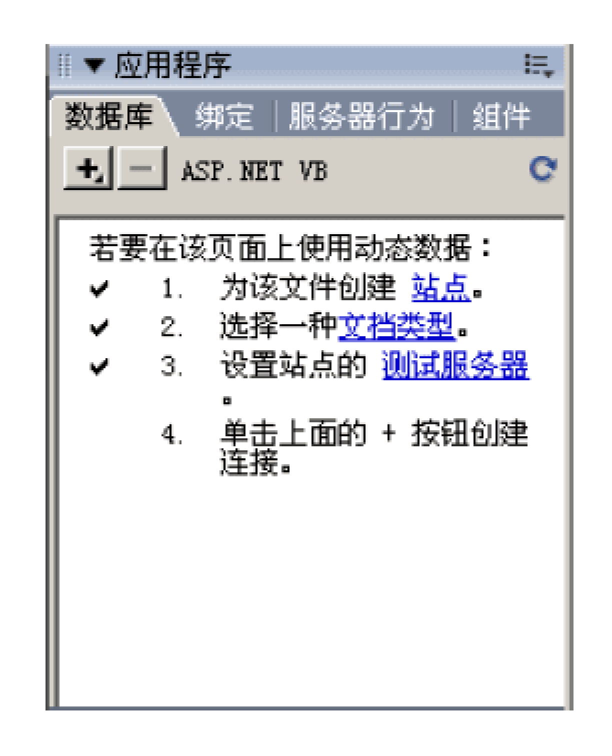


图 7.18　【数据库】面板

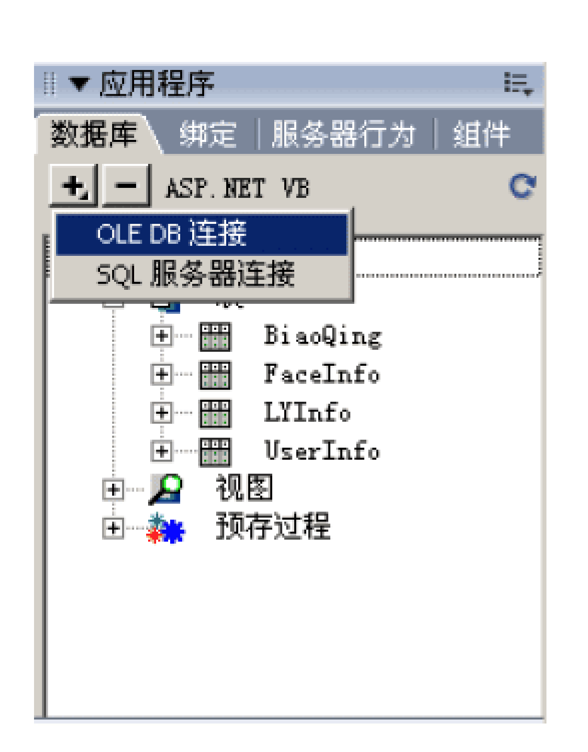


图 7.19　创建连接命令

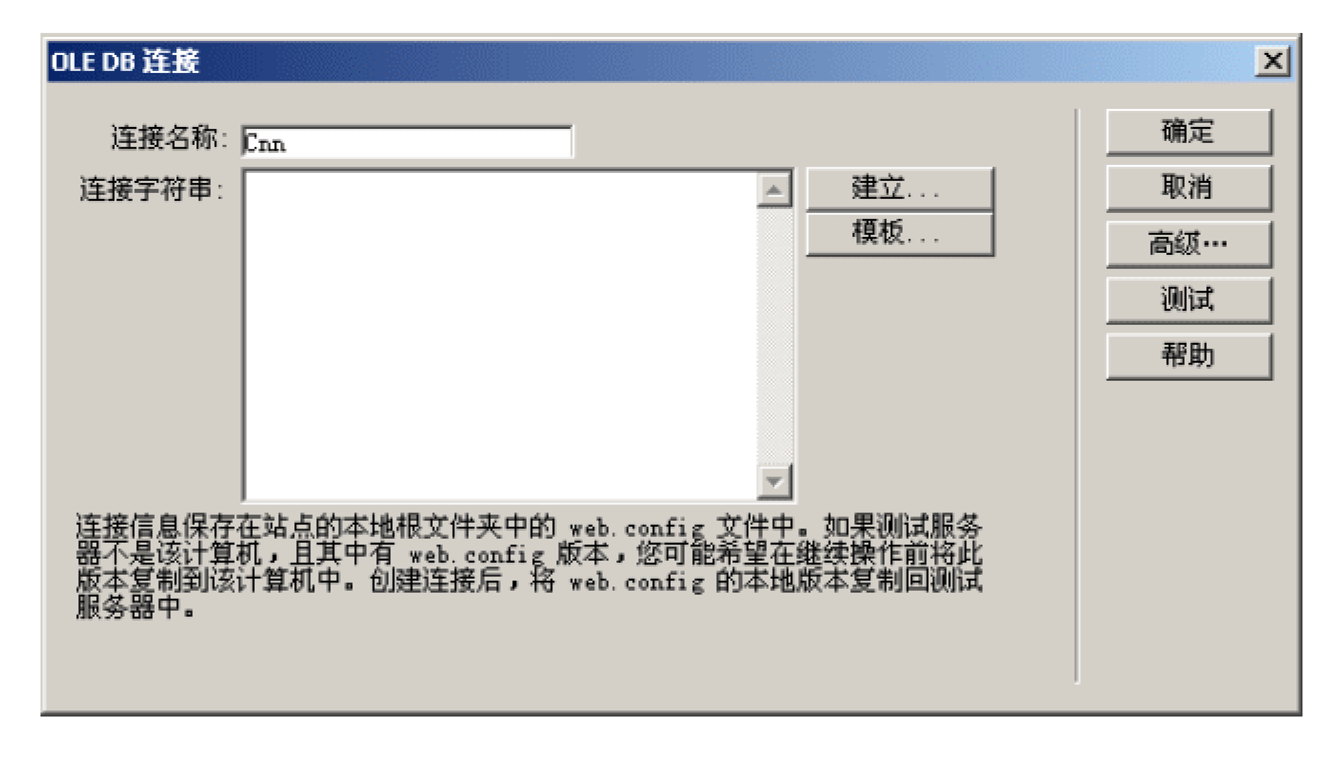


图 7.20　【OLE DB 连接】对话框

(3) 在连接名称中输入“Cnn”，然后单击【建立】按钮，此时将弹出 Data Link Properties 对话框。选择 Provider 选项卡，如图 7.21 所示。

在 OLE DB Provider 列表框中，选择 Microsoft Jet 4.0 OLE DB Provider 选项，然后单击

Next 按钮，此时将切换至 Connection 选项卡，如图 7.22 所示。通过单击【浏览】按钮 ...，选择所要连接的 Microsoft Access 数据库文件 data.mdb。单击 Test Connection 按钮，测试数据库的连接。如果连接成功，系统将弹出如图 7.23 所示的对话框。

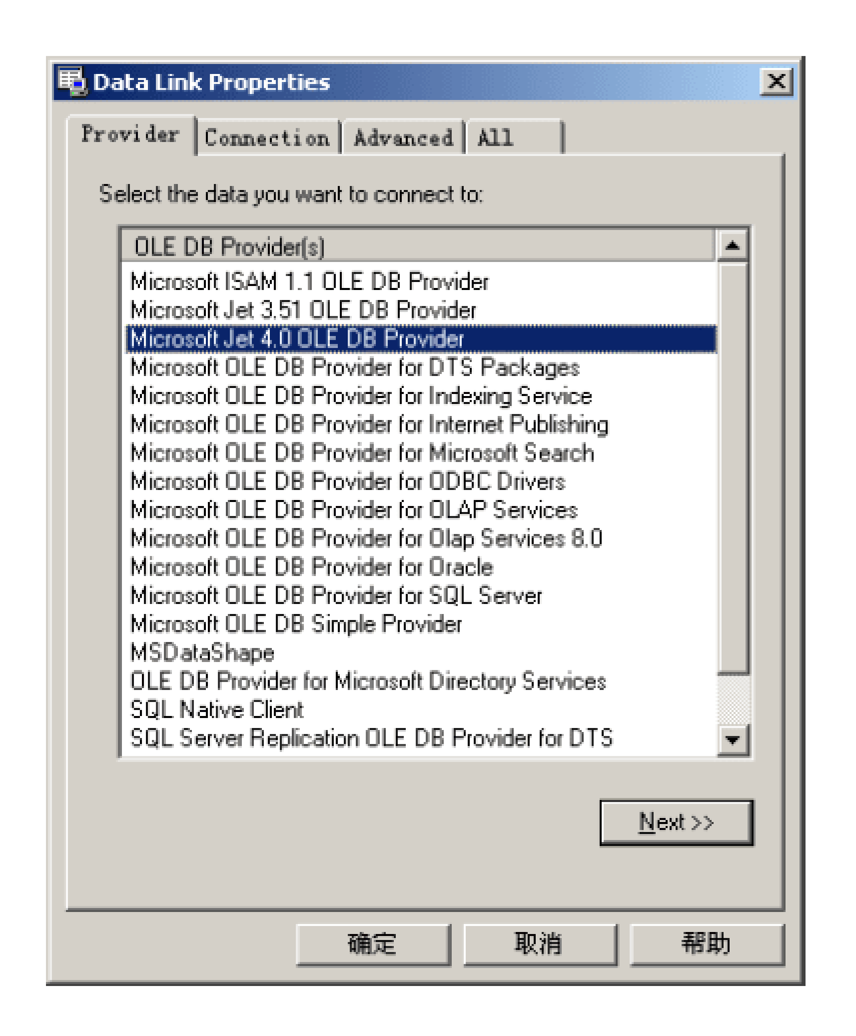


图 7.21　Provider 选项卡

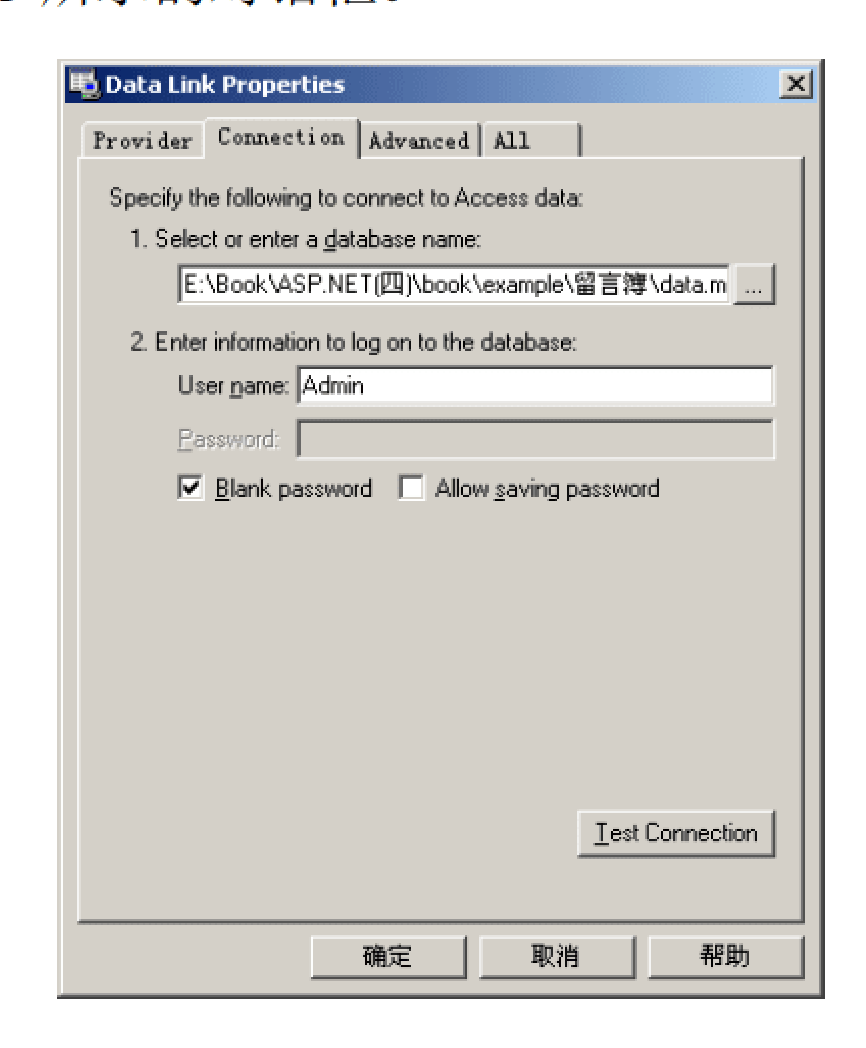


图 7.22　Conncetion 选项卡

(4) 单击【确定】按钮，返回【OLE DB 连接】对话框，此时在【连接字符串】文本框中将显示前面通过向导创建的数据库连接对应的字符串，如图 7.24 所示。

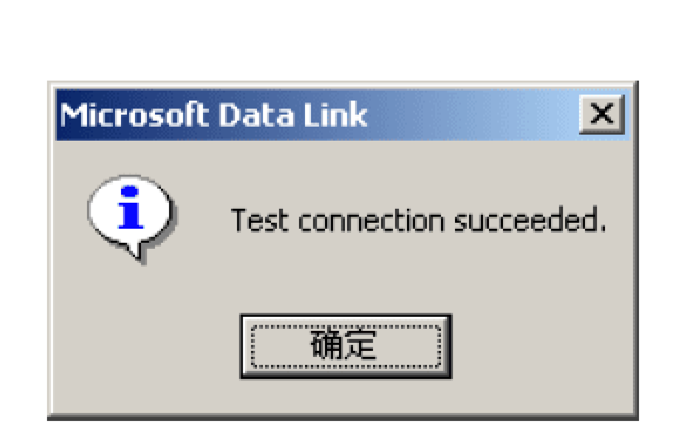


图 7.23　提示测试连接成功的对话框

图 7.24　【OLE DB 连接】对话框

(5) 单击【确定】按钮，完成数据库连接的创建。

至此，我们创建了一个连接至本系统数据库文件 Data.mdb 的数据库连接。此时，在【应用程序】面板组的【数据库】面板中，将会显示所创建的连接，如图 7.25 所示。

在所创建的名为 Cnn 的数据库连接下，显示了该数据库中的所有数据表、视图及预存过程。双击指定的数据表，可显示该数据表中定义的所有字段信息，包括字段名、数据类型以及字段大小等。

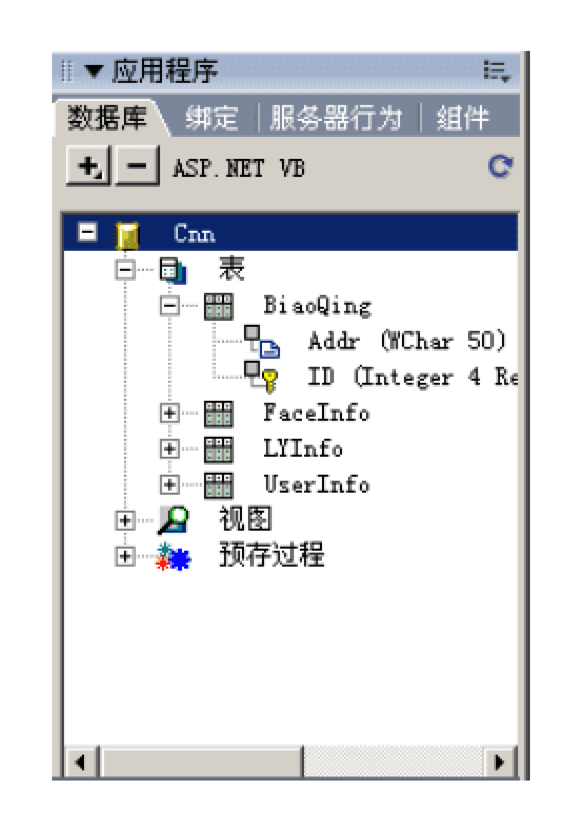


图 7.25　【数据库】面板

3. 创建数据集及数据的简单绑定

创建数据库连接之后，便可进行数据集的创建。所谓数据集，是指通过 SQL 语句查询数据库时返回的数据集合。创建

完数据集后便可将该数据集绑定至页面中，在绑定数据集中数据的地方动态地显示数据。

在本页面中，数据集的创建步骤如下。

(1) 打开【应用程序】面板组，将切换至【服务器行为】面板，如图 7.26 所示。

(2) 单击 + 按钮，系统将弹出一个菜单，其中包括【数据集】、【预存过程】、【数据网格】、【数据列表】、【重复区域】等命令，如图 7.27 所示。

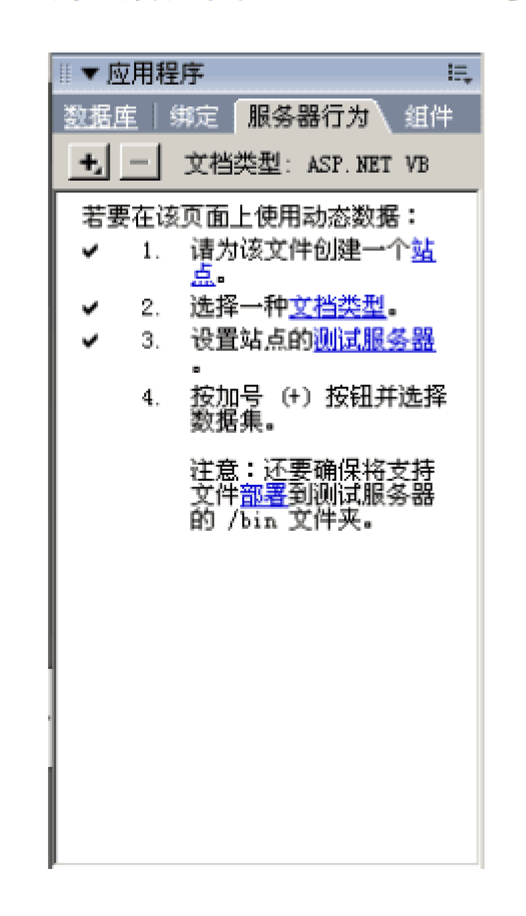


图 7.26　【服务器行为】面板

图 7.27　【服务器行为】菜单

(3) 选择【数据集】命令，系统将弹出【数据集】对话框，如图 7.28 所示。

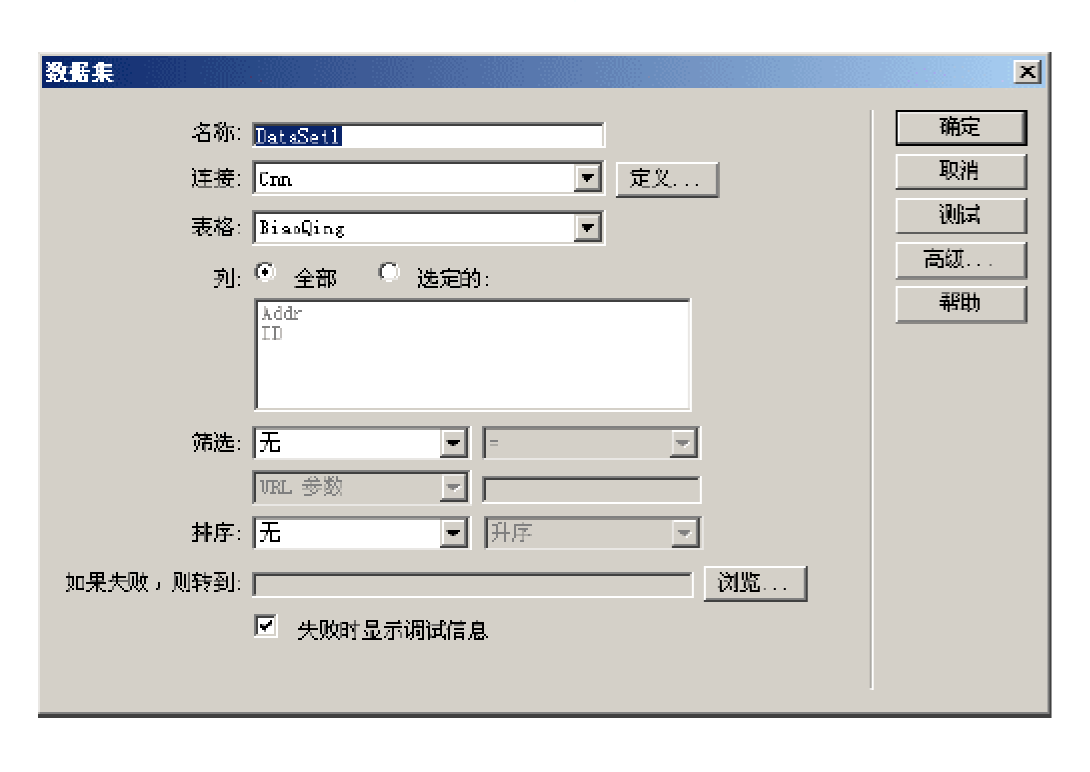


图 7.28　【数据集】对话框

(4) 在【连接】下拉列表中，选择前面定义的数据库连接 Cnn，此时在【表格】下拉列表中将会显示所连接的数据库中的所有表，选择相应的表，在下面的列表框中将会显示表中的所有字段。如果需要的数据集是一个数据表的所有记录，那么这里只需选择该表，单击【确定】按钮即可。但这里所要获取的留言信息是从 Lyinfo(留言信息表)、FaceInfo(头像信息表)、BiaoQing(表情信息表)3 个数据表中关联取出的，而不是从单个数据表中获取。单击【高级】按钮，将对话框由简单模式切换为高级模式，如图 7.29 所示。

(5) 在高级模式下可输入 SQL 语句，即可通过指定的 SQL 语句查询出所要返回的记录集。此处，输入以下 SQL 语句：

```
SELECT a.*,('~/pic/touxiang/'+b.picaddr) as
face_addr,('~/pic/biaoqing/'+c.addr) as bq_addr FROM lyinfo a,faceinfo
b,biaoqing c where faceid=b.id and biaoqingid=c.id order by a.id
```

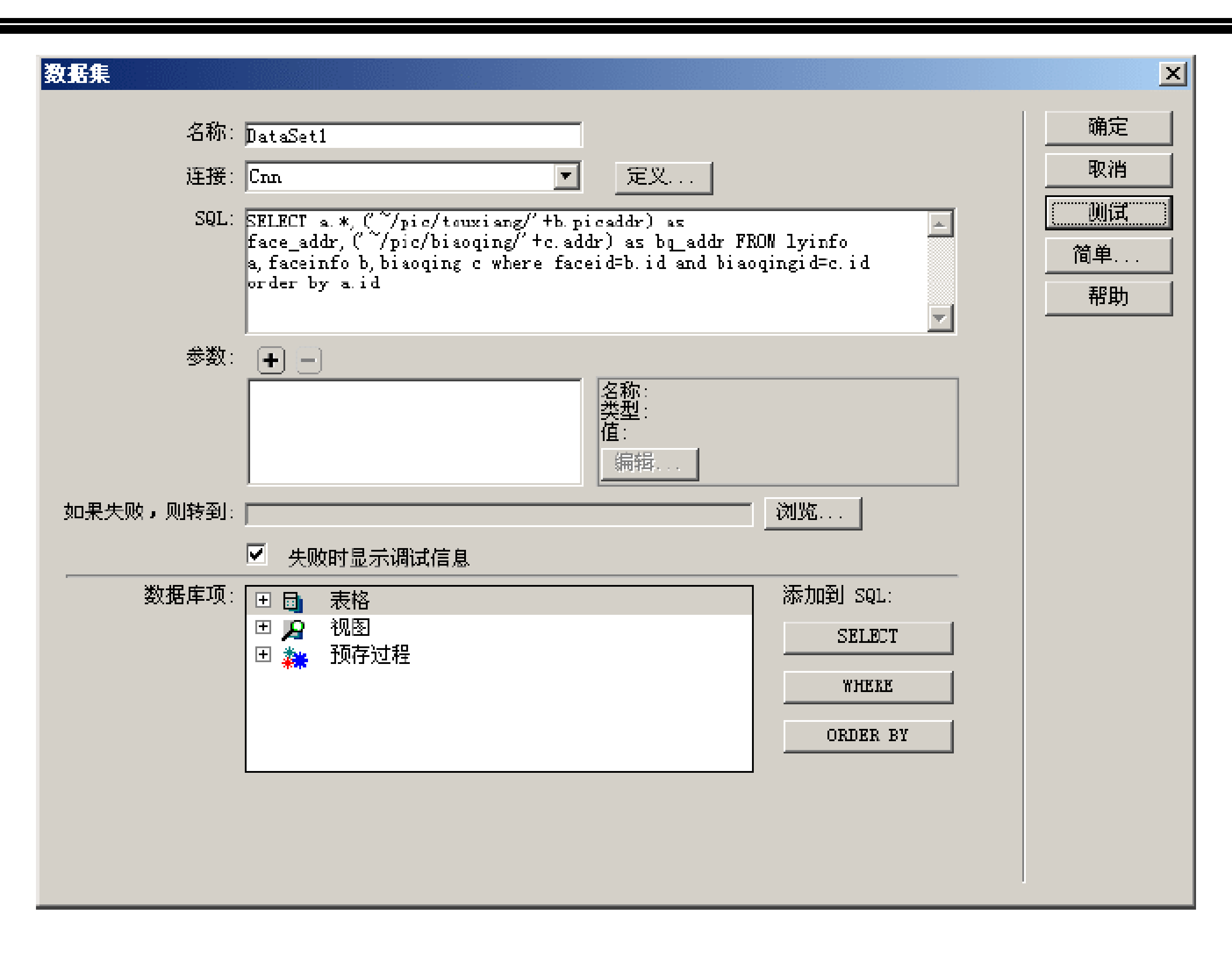


图 7.29　【数据集】对话框的高级模式

(6) 该语句将返回留言信息表的所有字段信息，以及个人头像和留言表情所对应的图片地址。在本系统中，个人头像对应的图片默认存放在 pic 目录下的 TouXiang 文件夹中，留言表情对应的图片默认存放在 pic 目录下的 BiaoQing 文件夹中，而在数据库中仅存放图片的文件名。

(7) 单击【测试】按钮，系统将运行输入的 SQL 语句，验证其正确性。如果正确，则显示查询结果，如图 7.30 所示。

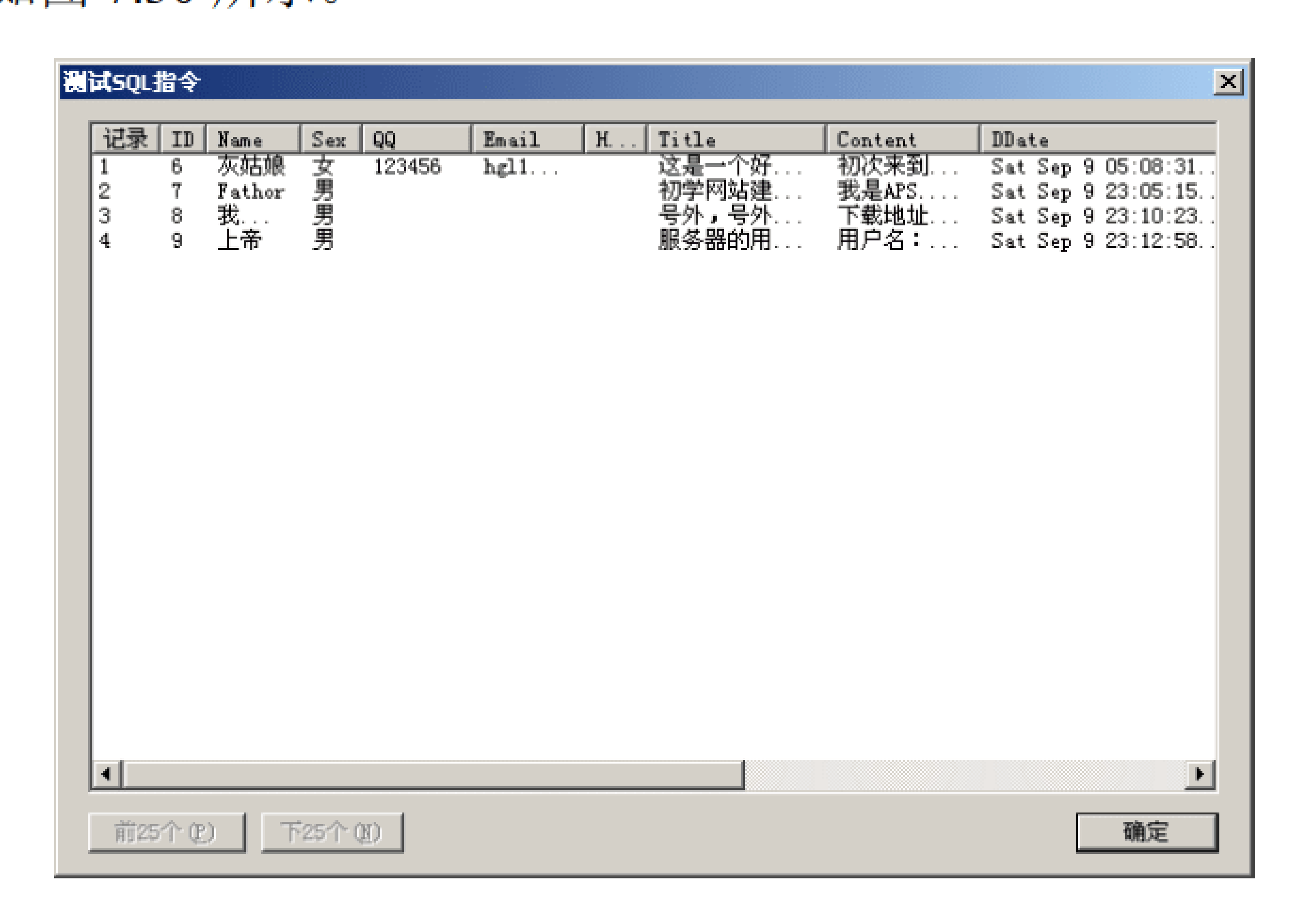


图 7.30　查询结果

(8) 单击【确定】按钮返回【数据集】对话框，再单击【确定】按钮即可完成数据集的创建。此时，在【应用程序】面板组的【服务器行为】面板中将会显示刚才创建的数据集 DataSet1，如图 7.31 所示。

(9) 接下来，可以将数据集中的数据绑定至已建好框架结构的页面中。打开【应用程序】面板组，切换至【绑定】面板，如图 7.32 所示。

(10) 在【绑定】面板中，显示了刚才创建的数据集，同时在数据集下还包含该数据集中的所有字段信息。直接将数据集中相应的字段拖至页面中相应的位置即可，如图 7.33 所示。

图 7.31　【服务器行为】面板

图 7.32　【绑定】面板

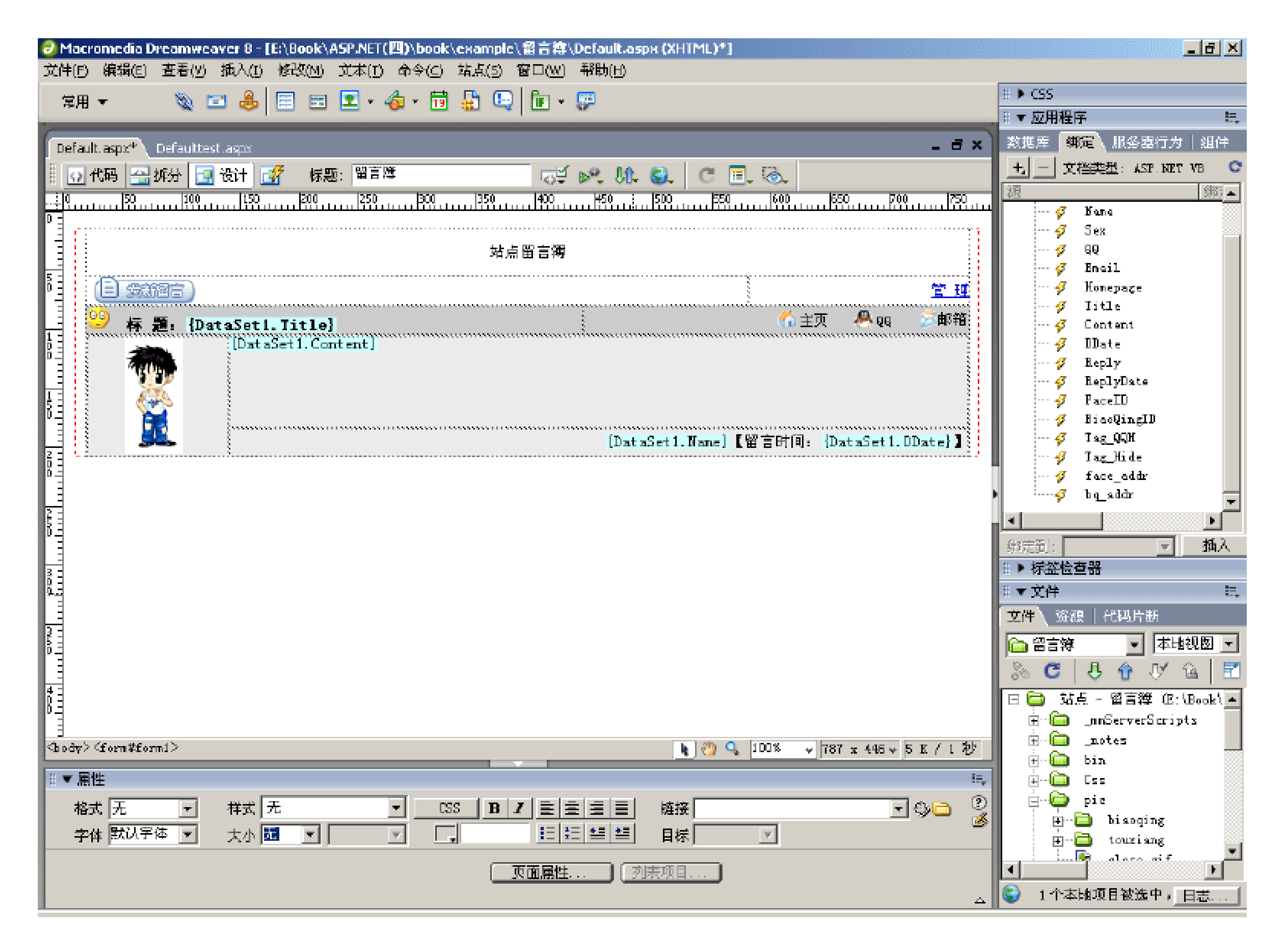


图 7.33　绑定数据后的页面

4. 动态数据的复杂绑定

此时，只是进行了初步的数据绑定，仍有大量的细化工作需要进行。

(1) 设置个人主页、QQ 及电子邮箱的相关链接。在【设计】视图中，选定“主页”图片，在【属性】面板中设置其链接属性的值为“<%# DataSet1.FieldValue("Home", Container) %>”；同理，分别设置 QQ 及电子邮箱图片的链接属性为“http://search.tencent.com/cgi-bin/friend/user_show_info?ln=<%# DataSet1.FieldValue("qq", Container) %>”和“mailto:<%# DataSet1. FieldValue("Email", Container) %>”。其中，表达式<%# DataSet1.FieldValue("字段名", Container) %>用于获取所绑定的数据集中指定字段的值。

(2) 选择留言标题前的留言表情图片，将视图切换至【代码】视图，将该 Image 服务器控件的 ImageUrl 属性的值设置为“<%# DataSet1.FieldValue("bq_addr", Container) %>”；将视图切换回【设计】视图，选择个人头像图片，再将视图切换至【代码】视图，将该 Image

服务器控件的 ImageUrl 属性的值设置为“<%# DataSet1.FieldValue("face_addr", Container) %>”。

现在，可以尝试预览该页面。如果页面运行在 ASP.NET 2.0 的环境下，那么可能会出现编译错误，如图 7.34 所示。

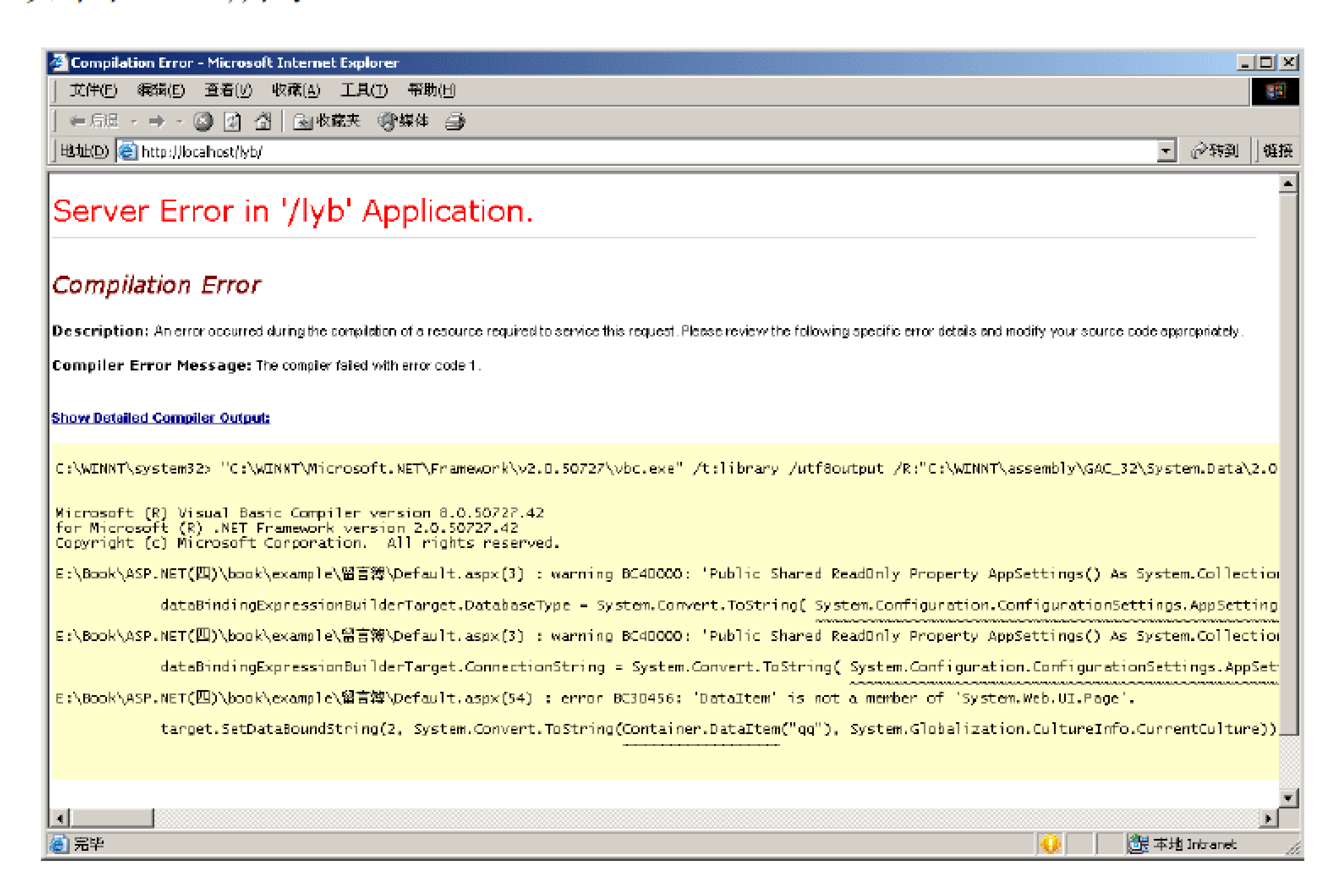


图 7.34　编译错误

这个问题的出现，是由于在 ASP.NET 2.0 中采用了 ConfigurationManager.AppSettings() 方法来替代 ASP.NET 1.0 中的 ConfigurationSettings.AppSettings()方法读取配置文件 Web.Config 中指定节点的值，而在 Dreamweaver 8 中生成的 ASP.NET 页面采用的仍是 ConfigurationSettings.AppSettings()方法。解决办法很简单，将页面切换至【代码】视图，使用 ConfigurationManager.AppSettings()替换所有的 ConfigurationSettings.AppSettings() 即可。

正常的预览效果，如图 7.35 所示。

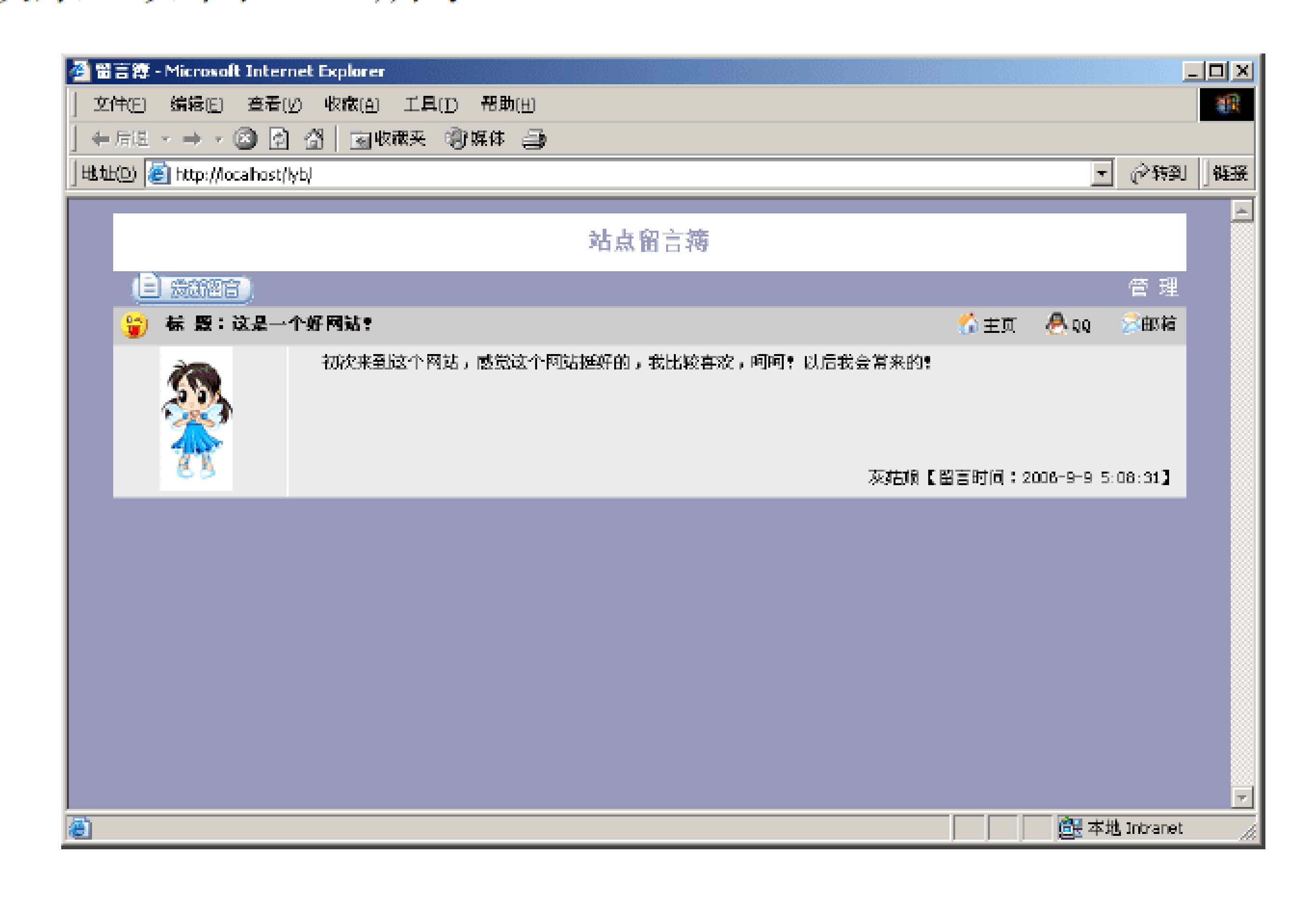


图 7.35　页面预览效果

5. 高级功能的添加

数据终于显示出来了，而且是从数据库动态取出的数据，但似乎还缺少点什么。不错，

还有几个问题没有解决：一是相关回复的显示；二是对留言信息的判断(包括该留言是否为悄悄话以及该留言是否已被屏蔽等)；三是这里仅显示了一条数据，而用户需要显示所有的留言信息。

下面，先来看一看多条数据的显示。

(1) 选定用于显示留言信息的表格 table3。打开【应用程序】面板组，切换至【服务器行为】面板，单击 按钮，系统将弹出一个菜单，如图 7.36 所示。

选择【重复区域】命令，此时将弹出【重复区域】对话框中，如图 7.37 所示。在【数据集】下拉列表框中选择前面定义的数据集 DataSet1，在【显示】单选按钮组中选择第一项，设置一次仅显示 10 条记录。

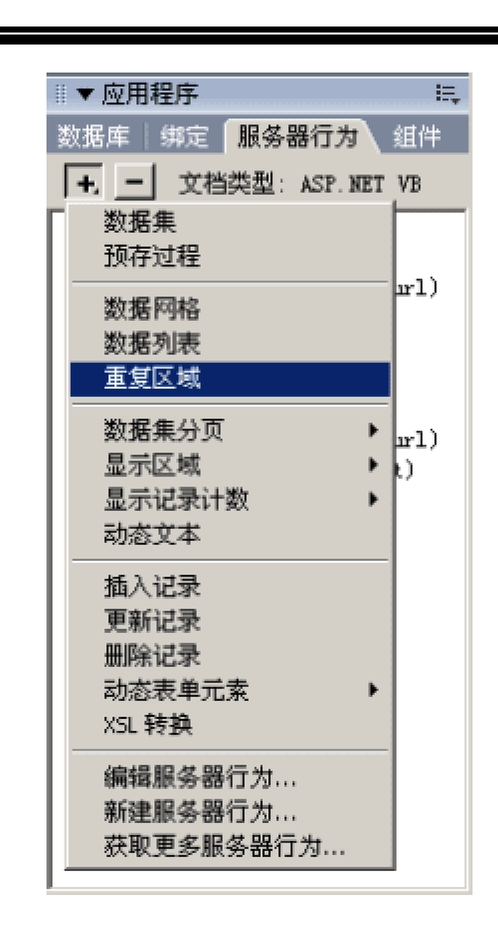


图 7.36　【服务器行为】菜单

图 7.37　【重复区域】对话框

单击【确定】按钮，将视图切换至【代码】视图，可以发现 Dreamweaver 8 在选定的表格标签外自动添加了一个 Repeater 控件，而整个表格代码均位于<ItemTemplate>和</ItemTemplate>标签之内，这就是要重复显示的区域。

(2) 再在不同的留言之间添加空行，以使页面更加美观。

将视图切换至【代码】视图，并将光标放置在</ItemTemplate>后。单击【ASP.NET-插入】工具栏中的 按钮，在弹出的【标签选择器】对话框中选择【ASP.NET 标签】|【模板】分类，并在右边的列表框中选择 SeparatorTemplate，如图 7.38 所示。

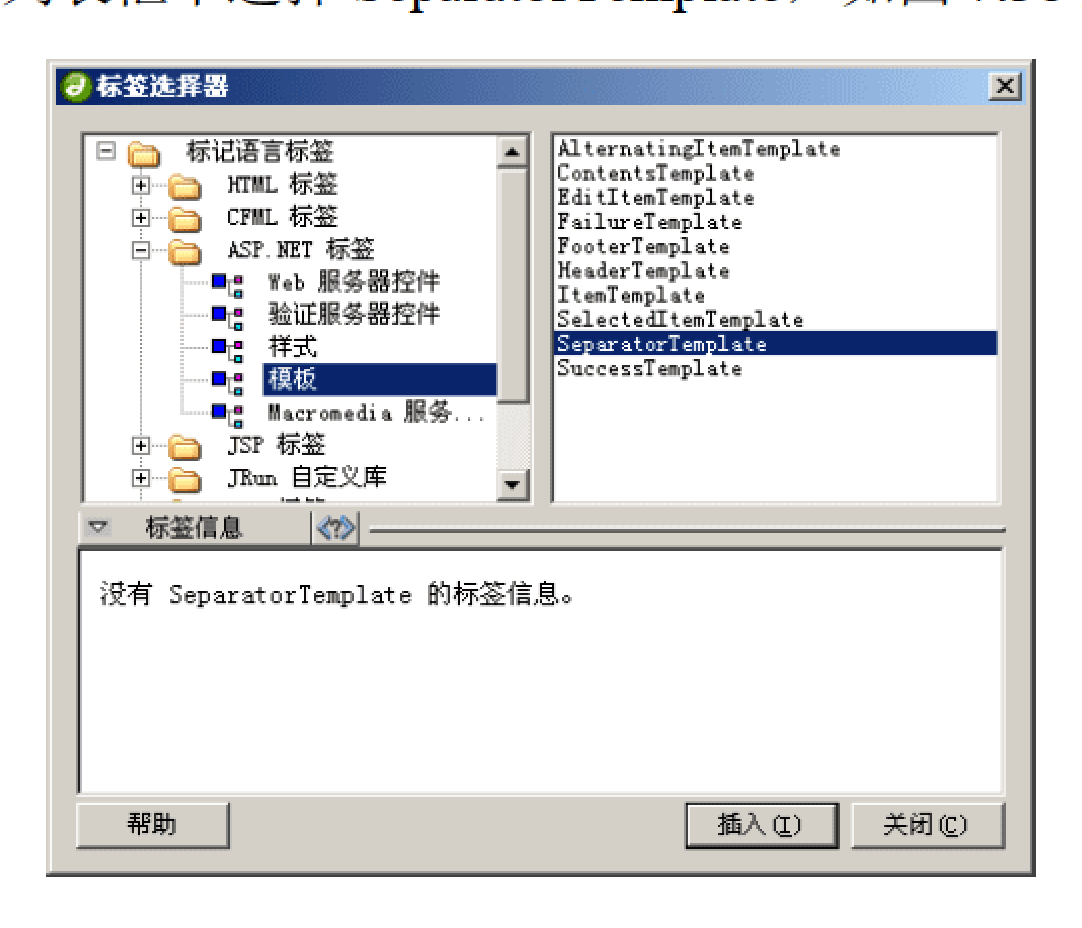


图 7.38　【标签选择器】对话框

单击【插入】按钮，此时在代码中将插入<separatortemplate>和</separatortemplate>标签对，在此标签对之间输入“
”，即空行，如图 7.39 所示。

图 7.39　代码示例

保存页面，再次在浏览器中浏览该页面，此时可以看到查询出来的留言信息的前 10 条，如图 7.40 所示。

图 7.40　页面预览

(3) 在代码中添加对留言信息相关回复，并对其是否为悄悄话及是否已屏蔽进行判断。这里需要提醒读者，虽然通过 Dreamweaver 8 快捷地创建 ASP.NET 页面，并实现了相关功能。但是，如果需要实现一些稍微高级的功能，则仍需对代码进行修改和调整。因此，掌握了 Dreamweaver 8 并不意味着学会了 ASP.NET，最重要的仍在于对 ASP.NET 基本语法的掌握。Dreamweaver 8 只是一种开发工具，它可以帮助设计界面，调整布局，但关键之处仍需要在其代码视图中编写 ASP.NET 代码。

将视图切换至【代码】视图，在<HTML>标记前添加 ASP.NET 脚本，如下所示：

【示例代码】

```
<script Language="VB" runat="server">
  dim n_count as Integer=0
</script>
```

这里，主要是定义一个整型变量 n_count，该变量用于存储记录集的当前行数。然后，将 Repeater 控件的相关代码修改如下。

【示例代码】

```
<ASP:Repeater runat="server" DataSource='<%# DataSet1.DefaultView %>'>
<ItemTemplate>
  <table width="750" border="0" align="center" cellspacing="0"
bgcolor="#EBEBEB">
  <tr>
```

```
    <td height="25" bgcolor="#cccccc" class="td1">
      <asp:Image ID="img_bq" ImageUrl=<%# dataset1.FieldValue("bq_addr",
Container) %> ImageAlign=AbsMiddle runat="server" /> <strong>标 题:
<%# dataset1.FieldValue("Title", Container) %></strong>
    </td>
    <td height="25" align="right" bgcolor="#cccccc" class="td2">
      <a href="<%# dataset1.FieldValue("Home", Container) %>"><img
src="pic/home.gif" width="44" height="16" border="0"
/></a>   
      <a
href="http://search.tencent.com/cgi-bin/friend/user_show_info?ln=<%#
dataset1.FieldValue("qq", Container) %>"><img src="pic/qq.gif" width="35"
height="16" border="0" /></a>   
      <a href="mailto:<%# dataset1.FieldValue("email", Container) %>"><img
src="pic/email.gif" width="42" height="16" border="0" /></a>
    </td>
  </tr>
  <tr>
    <td colspan="2">
    <table width="100%" border="0" cellpadding="0" cellspacing="0">
    <tr>
      <td width="120" rowspan="2" align="center" valign="middle"><asp:Image
ID="img_face" ImageUrl=<%# dataset1.FieldValue("face_addr", Container) %>
runat="server" /> </td>
      <td width=1 bgcolor="white" rowspan=2></td>
      <td height="78" valign="top">    
      <%
       '获取当前留言信息的 tag_hide 字段值，该字段表示当前留言是否被管理员屏蔽
Dim strHide As String =dataset1.defaultView.Table.Rows(n_count)("tag_hide")
'获取当前留言信息的 tag_hide 字段值，该字段表示当前留言是否为悄悄话
       Dim strQQH As String = dataset1.defaultView.Table.Rows(n_count)
("tag_qqh")
       Dim str_HF As String = ""
       '判断当前留言是否有回复信息，如果有回复，则将其赋值给变最 str_HF
       If Not IsDBNull(dataset1.defaultView.Table.Rows(n_count)("reply"))
Then
         str_HF = dataset1.defaultView.Table.Rows(n_count)("reply")
       End If
       If strHide = "1" Then
        '显示屏蔽信息
      %>
        此留言已被管理员屏蔽，^_^
      <% ElseIf strQQH = "1" Then '显示悄悄话信息 %>
        此留言为悄悄话，^_^
      <% Else '显示留言内容 %>
        <%# dataset1.FieldValue("Content", Container) %>
      <%
       End If
       If Not IsDBNull(str_HF) And str_HF <> "" Then
           '如果存在回复信息，则显示相关回复
      %>
```

```
        <br /><br />
        <div style="padding-left:70px;padding-top:5px;">
         <table width="450" bgcolor="white" border=0 cellpadding="0"
cellspacing="0">
         <tr>
          <td
style="padding-left:5px;padding-top:5px;padding-right:5px;padding-bottom:
5px;">
        <b>相关回复：</b><br /><%# dataset1.FieldValue("Reply", Container) %>
        </td>
        </tr>
        </table>
        </div>
      <%
       End If
      %>
      </td>
    </tr>
    <tr>
      <td align="right" class="td2"><%# dataset1.FieldValue("Name",
Container) %>【留言时间：<%# dataset1.FieldValue("DDate", Container) %>】</td>
    </tr>
    </table>
    </td>
  </tr>
  </table>
</ItemTemplate>
<separatortemplate>
  <br /><% n_count=n_count+1 %>
</separatortemplate>
</ASP:Repeater>
```

在以上代码中添加的 ASP.NET 代码段，用于控制留言信息及回复信息的显示。

① 通过语句 dataset1.defaultView.Table.Rows(n_count)("字段名")(其中，n_count 表示当前记录在记录集中的行数)分别获取 Tag_qqh(是否悄悄话标识)、Tag_hide(是否屏蔽标识)、Reply(回复信息)等字段的值。当 Tag_hide 的值为 1 时，表示当前留言已被管理员屏蔽，此时在留言内容区域显示文本“此留言已被管理员屏蔽，^_^”；当 Tag_qqh 的值为 1 时，表示当前留言为悄悄话，此时在留言内容区域显示文本“此留言为悄悄话，^_^”；当以上两种情况均不存在时，则在留言内容区域显示正常的留言内容。

② 当存在回复内容时，在留言内容的下方添加了一个<div>标签，并在其中添加了一个表格，用于显示回复内容。为区别于留言内容，表格的背景设置为白色。

至此，页面的功能实现已经全部完成，其浏览效果如图 7.41 所示。

现在，再次回过头来看代码。可以发现，页面中并没有导入调用 Microsoft Access 数据库所必须的 System.Data.OleDb 命名空间，也有没定义相关的数据库对象。所有的返回数据集代码均集成在以下代码中。

图 7.41　页面预览

【示例代码】

```
<MM:DataSet id="DataSet1" runat="Server" IsStoredProcedure="false"

ConnectionString='<%#System.Configuration.ConfigurationManager.AppSettin
gs("MM_CONNECTION_STRING_cnn1") %>'
    DatabaseType='<%#
System.Configuration.ConfigurationManager.AppSettings("MM_CONNECTION_DAT
ABASETYPE_cnn1") %>'
 CommandText='<%# "SELECT a.*,(" + Chr(39) + "~/pic/touxiang/" + Chr(39) +
"+b.picaddr) as face_addr,(" + Chr(39) + "~/pic/biaoqing/" + Chr(39) +
"+c.addr) as bq_addr FROM lyinfo a,faceinfo b,biaoqing c where faceid=b.id
and biaoqingid=c.id order by a.id" %>'
    Debug="true" PageSize="10" CurrentPage='<%#
IIf((Request.QueryString("DataSet1_CurrentPage") <>
Nothing),Request.QueryString("DataSet1_CurrentPage"), 0)  %>' >
</MM:DataSet>
```

同时，在页首引用了 Macromedia 公司开发的用户控件 DreamweaverCtrls，如下：

```
<%@ Register TagPrefix="MM" Namespace="DreamweaverCtrls"
Assembly="DreamweaverCtrls,version=1.0.0.0,publicKeyToken=836f606ede05d4
6a,culture=neutral" %>
```

也就是说，在使用 Dreamweaver 8 开发 ASP.NET 应用程序时，在 ASP.NET 页面看到的只是用户控件 DreamweaverCtrls 的一些属性描述代码，而不是 VB.NET 代码，这些代码均已被编译在 DreamweaverCtrls.dll 文件中了。

对于初学者来说，这是一种快捷的途径，可以迅速开发出 ASP.NET 应用程序，而无须过多了解代码实现的原理。

7.2.2　添加留言页面

通过添加留言页面，用户可发表新的留言信息。下面，来看一看添加留言页面的实现。

在浏览留言页面的功能实现中，详细介绍了页面框架结构和系统功能的具体实现。在本页面及以后的功能介绍中，将忽略页面布局的具体操作，重点介绍页面功能的实现。

1. 添加控件

(1) 设计页面，如图 7.42 所示。

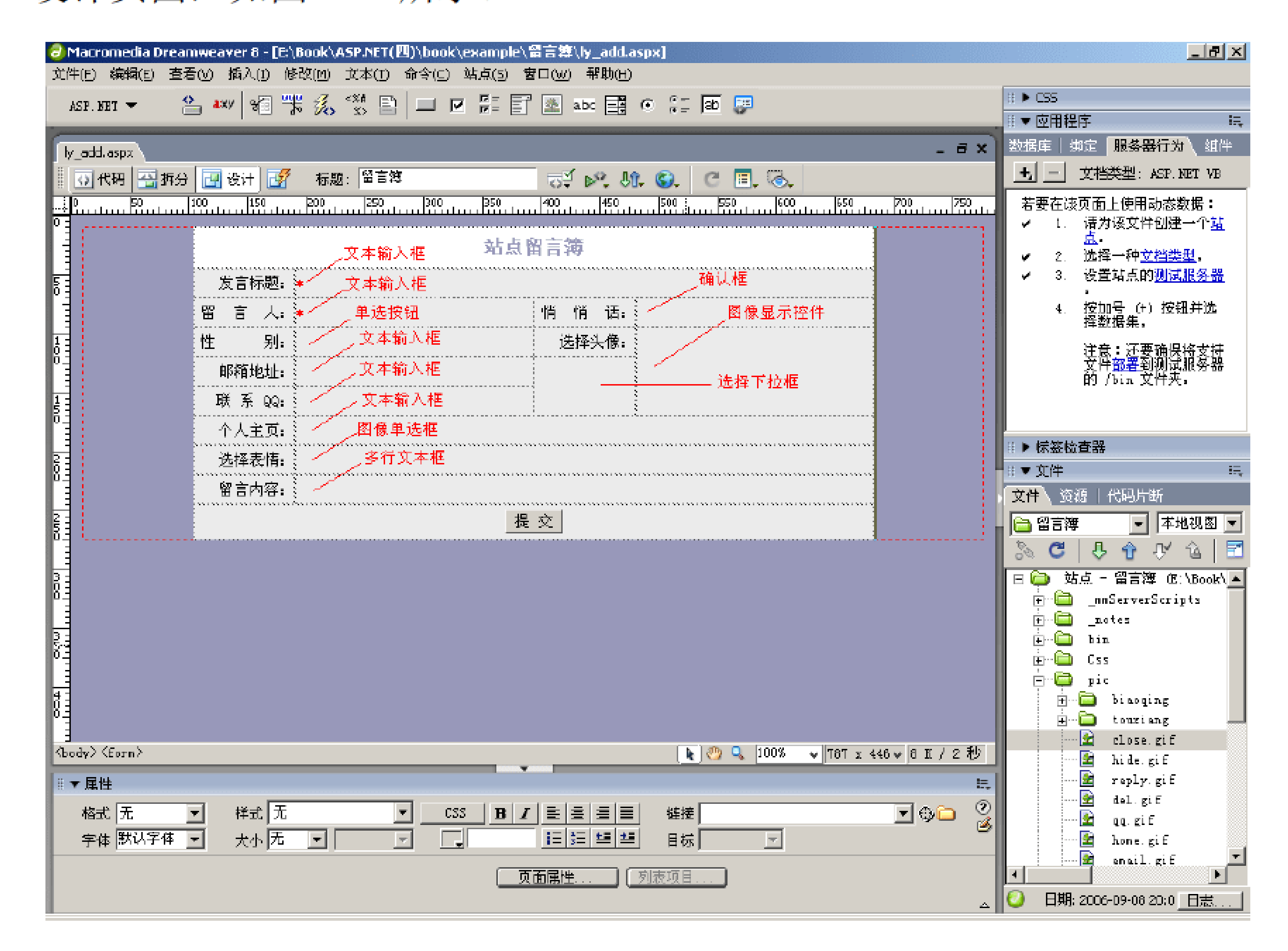


图 7.42　页面布局

在页面中，添加了一个窗体，并在窗体中添加了一个表格，同时设置了相应的文本信息。在图中，用文字标记了各信息输入单元格中所需放置的 ASP.NET 服务器控件。下面向其中添加控件。

在【ASP.NET-插入】工具栏中，提供了常用的 ASP.NET 服务器控件，如图 7.43 所示。

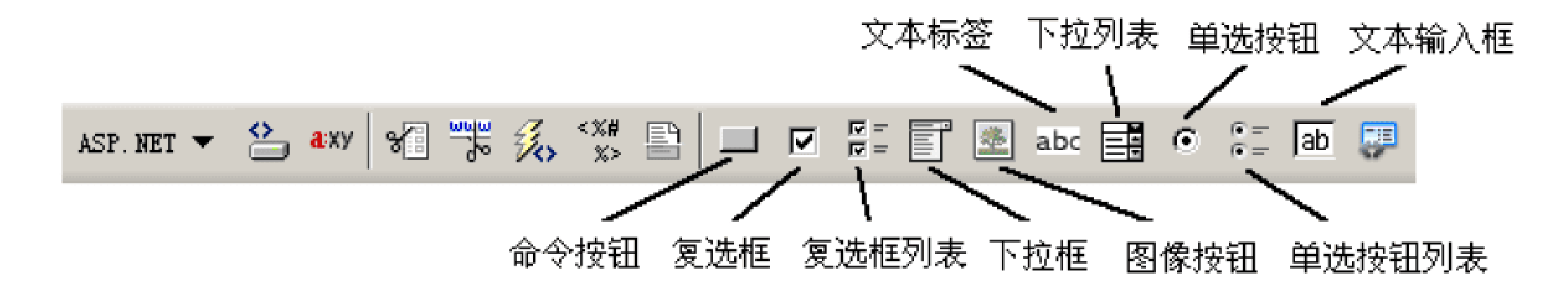


图 7.43　【ASP.NET-插入】工具栏

(2) 将【ASP.NET-插入】工具栏中的相应控件按钮拖至页面相应的位置，即可添加该控件。拖动【文本框】按钮至文字“发言标题”后的单元格中，此时将弹出【asp:文本框】对话框，如图 7.44 所示。

在该对话框中设置 ID 为 title，【文本模式】为【单行】，单击【确定】按钮即可。与

此类似，添加留言人、邮箱地址、联系 QQ、个人主页及留言内容等对应的文本框控件，对应的控件 ID 分别为 Lyr、E-mail、QQCode、HomePage、Content。其中，留言内容对应的文本框控件的模式为多行，列数设为 54，行数设为 5，如图 7.45 所示。

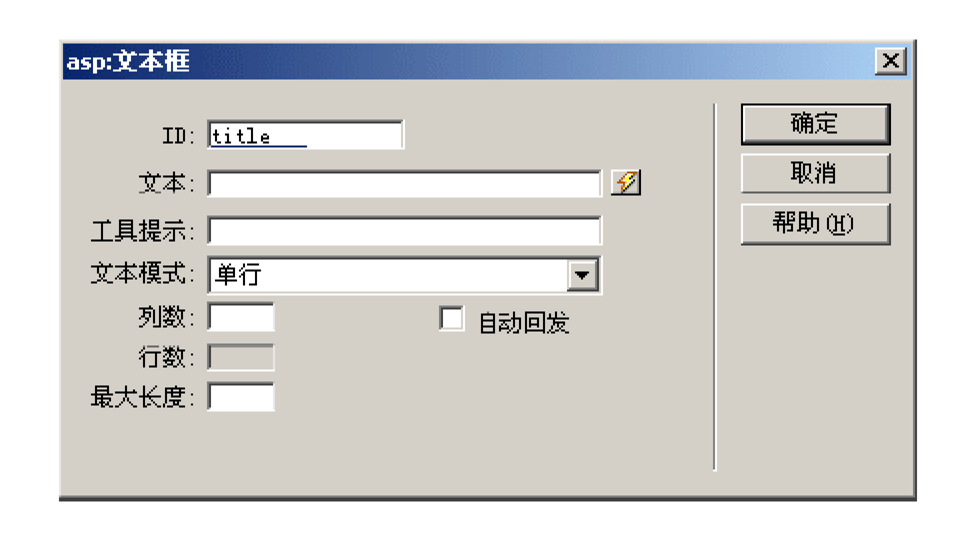


图 7.44　【asp:文本框】对话框

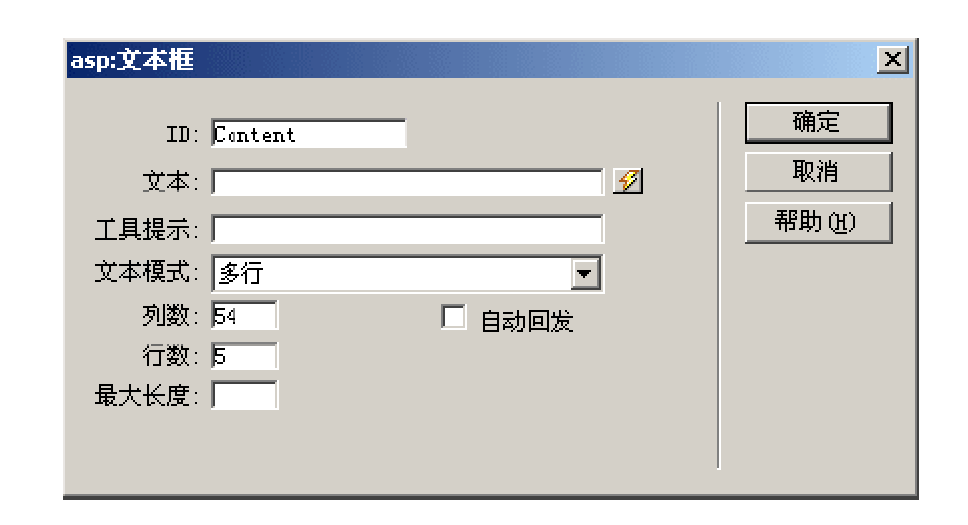


图 7.45　留言内容对应的文本框控件的设置

(3) 拖动复选框控件至文字“悄悄话”后面的单元格中，此时将弹出【asp:复选框】对话框，设置 ID 为 QQH，【文本】为“此留言为悄悄话”，【文本对齐】为【右对齐】，如图 7.46 所示。单击【确定】按钮，即可添加该复选框控件。

(4) 拖动单选按钮控件至文字“性别”后面的单元格中，此时将弹出【asp:单选按钮】对话框，设置其 ID 为 Radio1，文本为“帅哥”，【文本对齐】为【右对齐】，如图 7.47 所示。

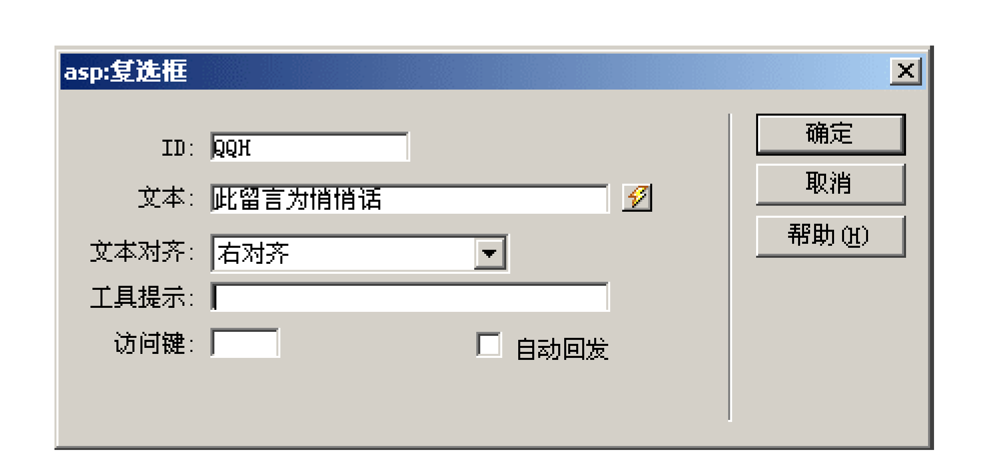


图 7.46　【asp:复选框】对话框

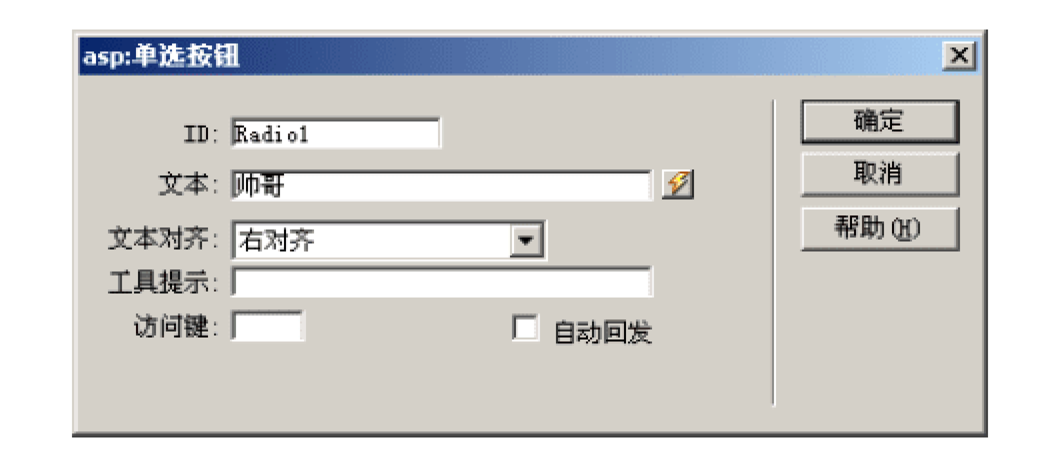


图 7.47　【asp: 单选按钮】对话框

单击【确定】按钮，即可添加该单选按钮控件。选择该控件，在【属性】面板中设置其选中状态为“已选中”，【组名称】为 Group1。与此类似，再次添加一个单选按钮控件，其 ID 为 Radio2，【文本】为“靓女”，选中状态为“未选中”，【组名称】为 Group2。

提示： 通过将多个单选按钮的组名称设为相同，可实现惟一选择。即选中其中一个单选按钮，其他单选按钮的状态将自动切换为“未选中”。

(5) 将光标置于文字“选择头像”后的单元格中，单击按钮，在弹出的【标签选择器】对话框中选择【ASP.NET 标签】分类，在右边的列表框中选择 asp:Image，插入一个 Image 控件，设置其 ID 为 DispImg，高度为 100 像素，宽度为 50 像素，替换文本为“头像预览”，其图像 URL 暂不设置，在程序中由代码绑定。

(6) 在如图 7.42 所示的页面框架中，“选择头像”下面的单元格中需添加一个下拉列表框控件。该控件用于显示从数据库中获取的所有头像名称以供用户选择，选择不同的头像名称，右侧的 Image 控件将显示相应的预览图像。为此，需创建一个数据集，并将其中的数据绑定到该下拉列表框控件，以便实现头像名称的动态显示。

打开【应用程序】面板组，切换至【服务器行为】面板，单击按钮，在弹出的菜单

中选择【数据集】命令，将弹出【数据集】对话框，如图 7.48 所示。

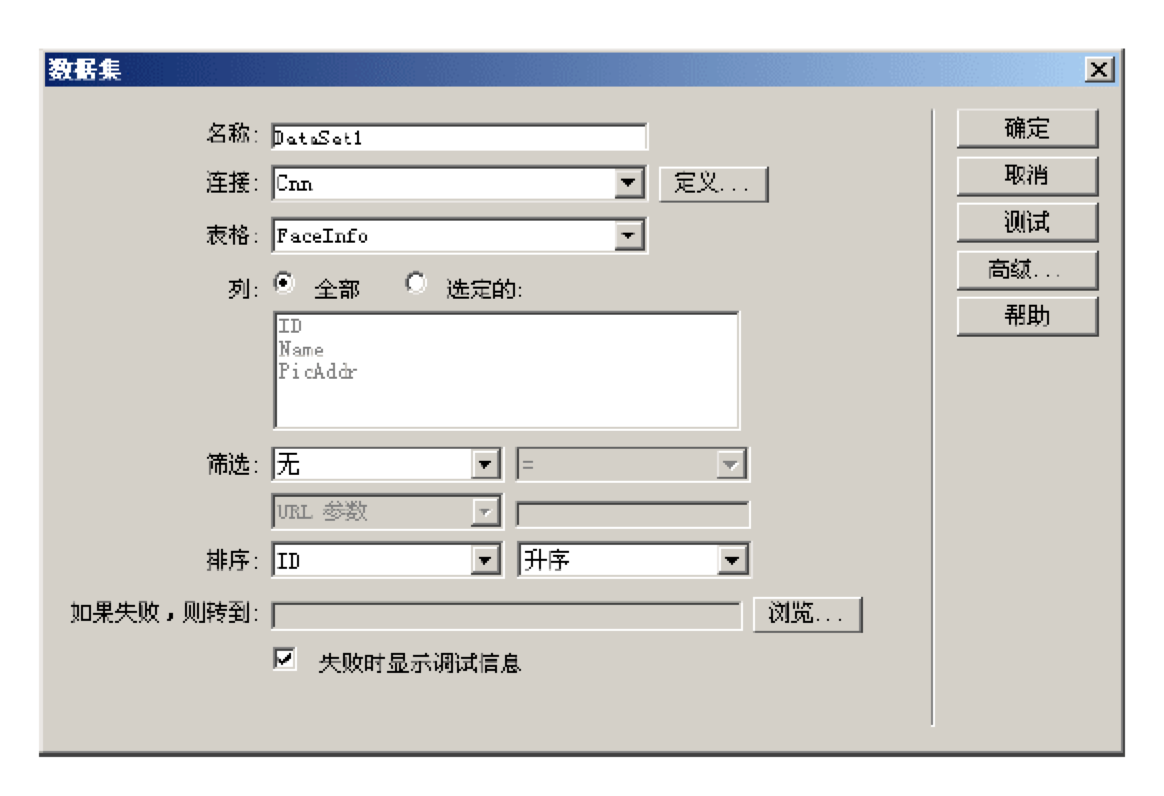


图 7.48　【数据集】对话框

在对话框中，设置数据集名称为 DataSet1，选择前面所定义的连接 Cnn，在【表格】下拉列表框中选择 FaceInfo，排序方式设为按 ID 升序排序。单击【确定】按钮，完成数据集的创建。

(7) 拖动下拉列表框控件至文字“选择头像”下的单元格中，此时将弹出【asp:下拉列表】对话框，如图 7.49 所示。

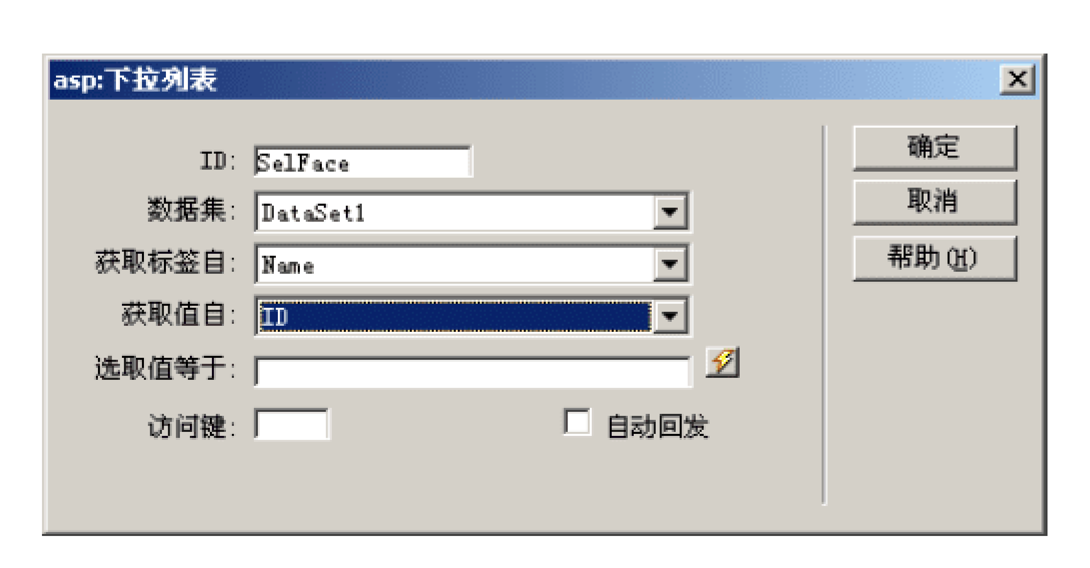


图 7.49　设置下拉列表 SelFace

将 ID 设置为“SelFace”，【数据集】选项设为 DataSet1，【获取标签自】选项(即显示名称)设为数据集中的字段 Name，【获取值自】(即返回值)选项设为数据集中的字段 ID，单击【确定】按钮。

(8) 此外，考虑到当下拉列表框控件 SelFace 的值发生改变时，除了需要获取其对应的 ID 值(该值保存至数据库中)，还需获取该选项对应的图像地址，即字段 PicAddr。如果在获取 ID 后再从数据库中读取其对应的字段 PicAddr 的值，那么在操作中将会出现多次与数据库交互，就降低了程序的效率。为此，在下拉列表框控件 SelFace 后再添加一个隐藏的下拉列表框控件，其 ID 为 SelFace1，【数据集】仍选择 DataSet1，【获取标签】为字段 Name，【获取值自】为字段 PicAddr，如图 7.50 所示。

选择该控件并右击，从弹出的子菜单中选择【编辑标签】命令，在弹出的【标签编辑器-DropDownList】对话框中选择【样式信息】分类，取消对【可见】复选框的选择，如图 7.51 所示。

单击【确定】按钮，即可将下拉列表框控件 SelFace1 设置为隐藏。

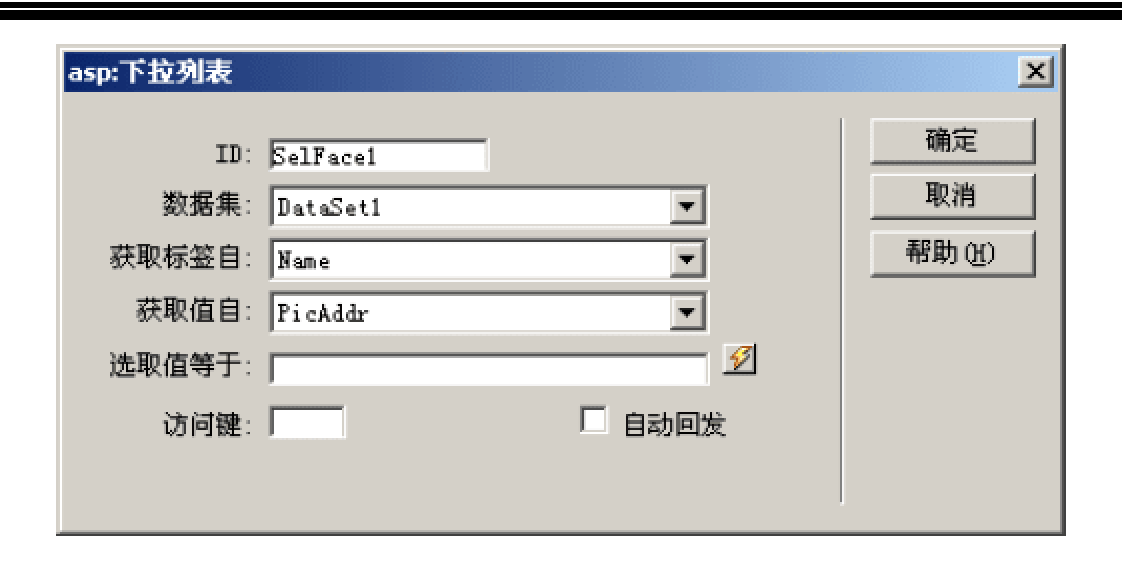


图 7.50　设置下拉列表 SelFace1

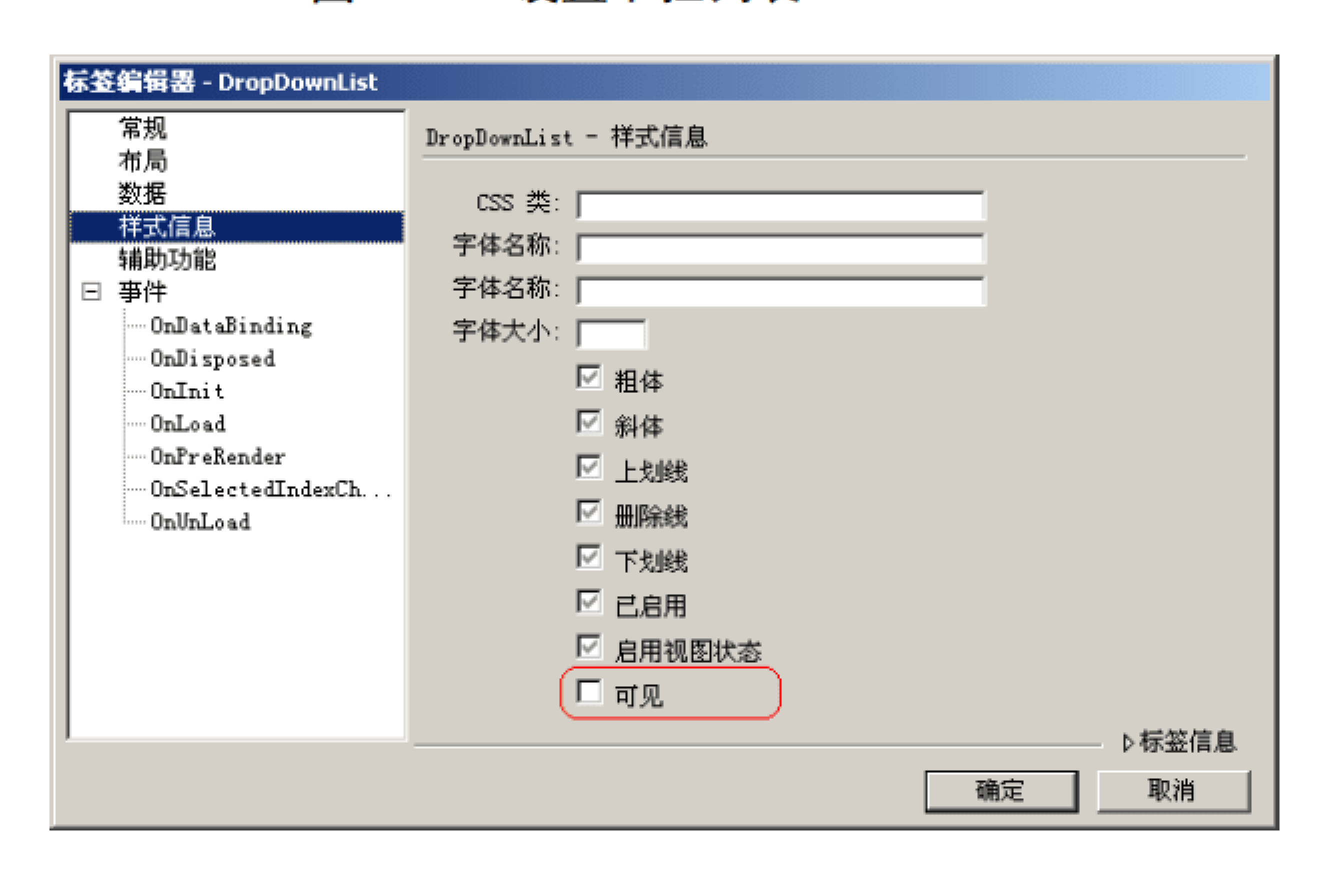


图 7.51　【标签编辑器-DropDownList】对话框

(9) 对于文字“选择表情”对应的图像单选按钮，通过 DataList 控件来实现。为此，需要再创建一个数据集，将该数据集命名为 DataSet2，如图 7.52 所示。

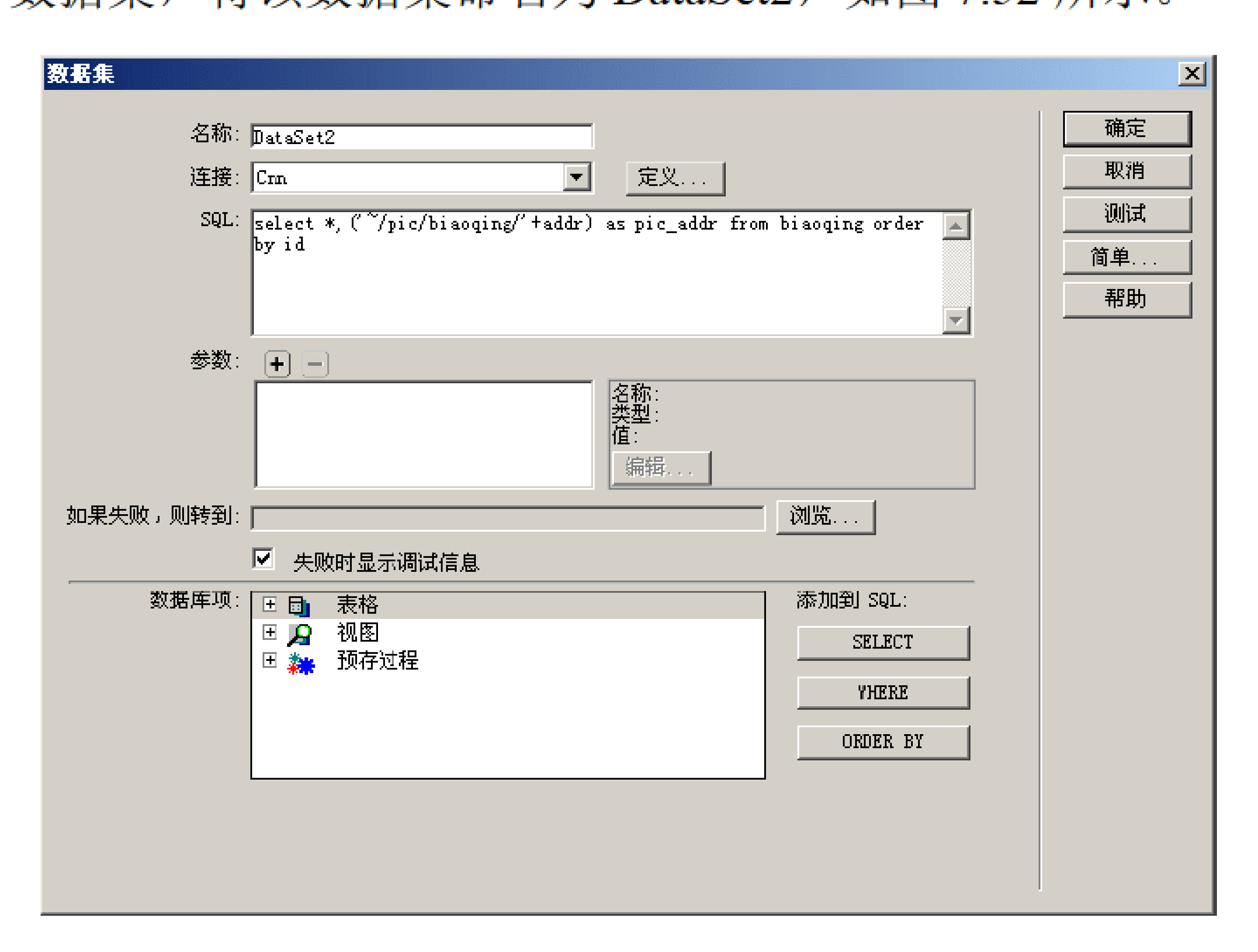


图 7.52　创建数据集 DataSet2

由于这里需要取得表情对应的图片地址，因此把【数据集】对话框切换到高级模式，输入 SQL 语句，如下：

```
select *,('~/pic/biaoqing/'+addr) as pic_addr from biaoqing order by id
```

(10) 定义数据集之后，打开【应用程序】面板组，切换至【服务器行为】面板，单击

按钮，在弹出的菜单中选择【数据列表】命令，此时将弹出【数据列表】对话框，如图 7.53 所示。

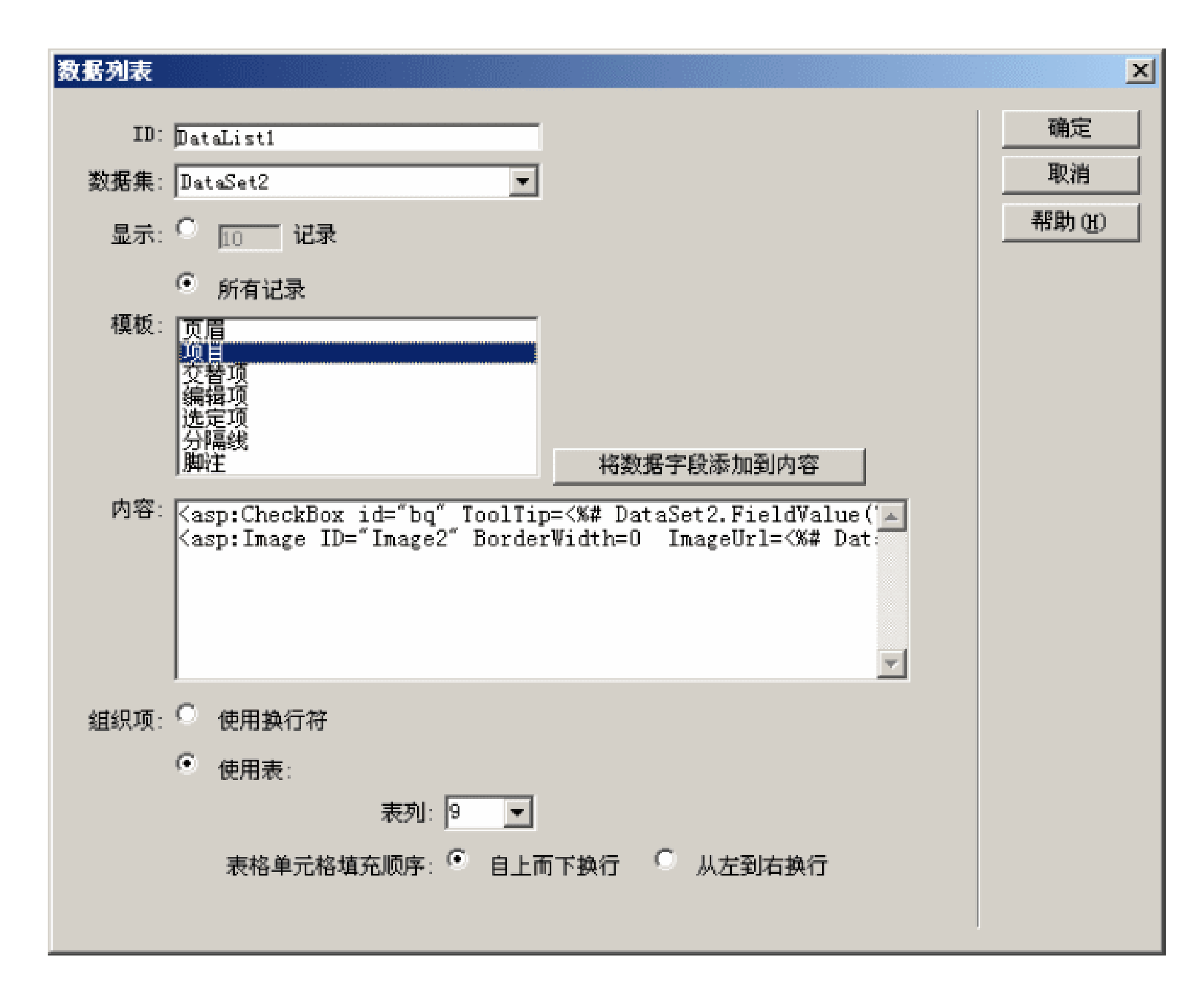


图 7.53　【数据列表】对话框

在对话框中，将 ID 设置为 DataList1，在【数据集】下拉列表框中选择 DataSet2，【显示】选项中选择【所有记录】单选按钮，【组织项】选项中选择【使用表】单选按钮，同时将【表列】设置为 9，选择【自上而下换行】单选按钮。在【模板】列表框中，选择【项目】选项，并在【内容】文本框中输入：

```
<asp:CheckBox id="bq" ToolTip=<%# DataSet2.FieldValue("ID", Container) %>
Checked="false"
OnCheckedChanged="Check_Changed"
AutoPostBack=true
runat="server" />
<asp:Image ID="Image2"
BorderWidth=0
ImageUrl=<%# DataSet2.FieldValue("pic_addr", Container) %>
runat="server" />
```

这里添加了一个复选框控件和一个 Image 控件，作为重复显示的内容。其中，复选框控件的 ToolTip 属性设置为数据集 DataSet2 中的字段 ID 的值，AutoPostBack 属性为 True，OnCheckChanged 事件为 Check_Changed，这样当列表中其中一个复选框被选中时，将会自动触发自定义的 Check_Changed 事件。而在所定义的 Image 控件中，将其 ImageUrl 属性设置为数据集 DataSet2 中的字段 Pic_addr 的值。

至此，界面设计基本完成，设计视图如图 7.54 所示。

2. 添加事件代码

下面，来添加相关的事件代码。

(1) 将视图切换至【代码】视图，将光标置于<HTML>标记前，单击【ASP.NET−插入】工具栏中的按钮，系统将在光标所在处插入一段 ASP.NET 代码，如下所示：

```
<script runat="server">
Sub Page_Load(Src As Object, E As EventArgs)
If Not IsPostBack Then
DataBind()
End If
End Sub
</script>
```

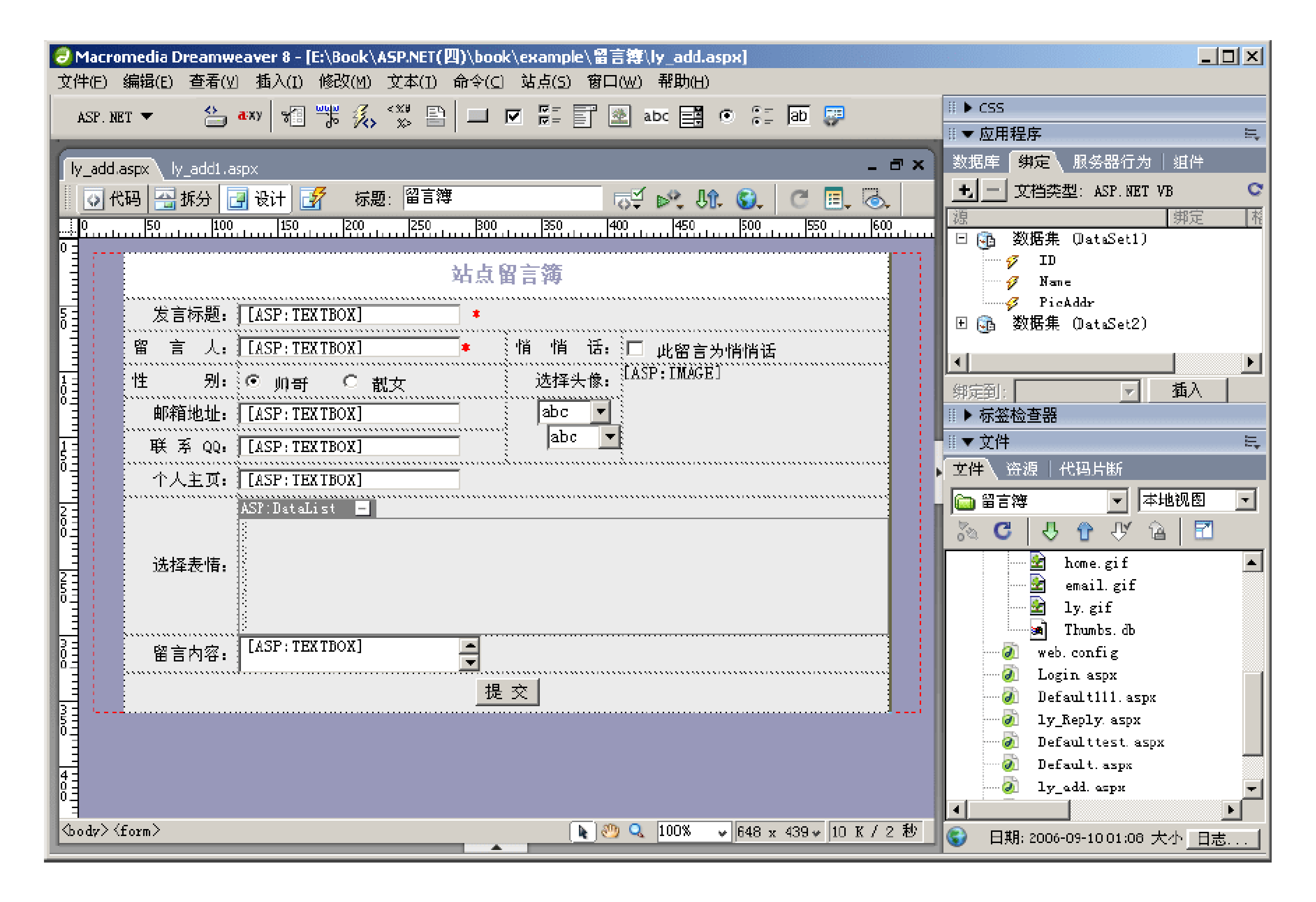


图 7.54　页面框架

这段代码定义了 Page_Load 事件(该事件在页面加载时执行)，并在其中调用了一个默认的过程 DataBind()。该过程通常用于绑定数据，这里可以将其删除。由于需要设置显示头像对应的 Image 控件的初始值，因此可在 Page_Load 事件中添加语句：

```
dispimg.ImageUrl="~/pic/touxiang/" & dataset1.defaultView.Table.Rows(0)
("picaddr")
```

在此语句中，通过获取数据集 DataSet1 第一条记录中的 PicAddr 字段的值(即头像对应的图片名称)，并加上图片地址前缀，然后赋值给 Image 控件的 ImageUrl 属性。这样，可避免当页面初次加载且用户尚未选择头像时 Image 控件无法正常显示。

(2)　在 Page_Load 事件代码后，添加名为 Check_Changed 的事件代码，如下：

【示例代码】

```
Sub Check_Changed(ByVal Sender As Object, ByVal E As EventArgs)
  '定义两个 CheckBox 控件类型变量
  Dim ObjCHK As CheckBox
  Dim ObjCHK1 As CheckBox
  Dim i As Integer
  '获取触发当前事件的 CheckBox 控件，并赋给 ObjCHK 变量
  ObjCHK = CType(Sender, CheckBox)
  If ObjCHK.Checked Then
```

```
    '当 ObjCHK 变量所表示的 CheckBox 控件为选中状态时，取消数据列表内其他 CheckBox
控件的选中状态
      For i = 0 To DataList1.Items.Count - 1
        ObjCHK1 = DataList1.Items(i).FindControl("bq")
        If ObjCHK1.ToolTip <> ObjCHK.ToolTip And ObjCHK1.Checked Then
          ObjCHK1.Checked = False
        End If
      Next
    End If
  End Sub
```

该事件代码在数据列表中的 CheckBox 控件的选中状态发生改变时触发。在代码中，首先获取数据列表中触发当前事件的源，即 CheckBox 控件，如果该控件的状态为选中状态，则将数据列表中的其余 CheckBox 控件置为未选中状态。

(3) 添加当下拉列表框 SelFace(选择头像)的选项发生改变时所触发的事件。首先，选择下拉列表框控件 SelFace 并右击，从弹出的子菜单中选择【编辑标签】命令，在弹出的【标签编辑器-DropDownList】对话框中的【常规】分类中选中【自动回发】复选框。然后，选择【事件】|【OnSelectedIndexChanged】分类，在右边的文本框中输入 Index_Changed(这是触发事件的名称)，如图 7.55 所示。

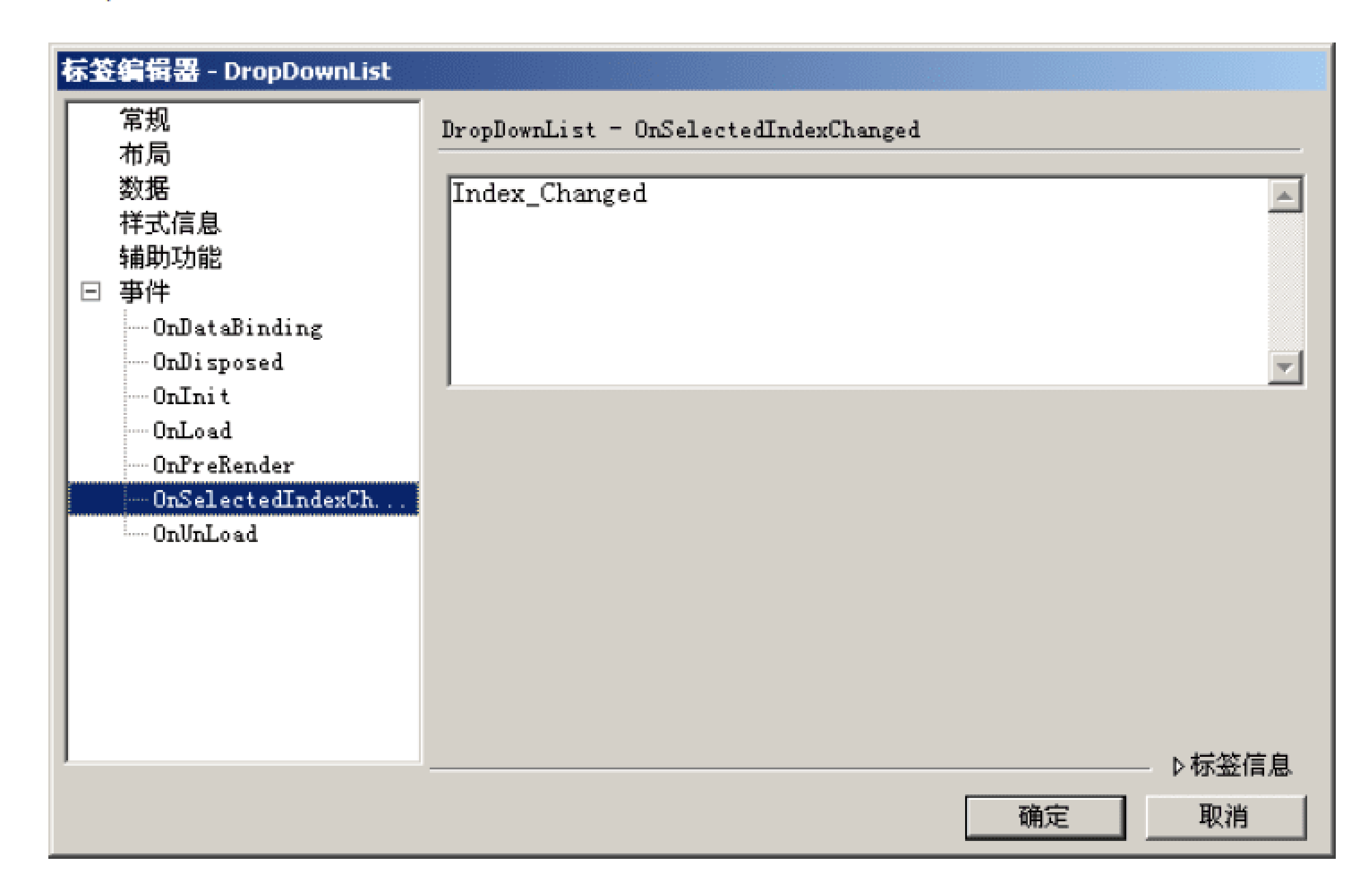


图 7.55　【标签编辑器-DropDownList】对话框

(4) 单击【确定】按钮，将视图切换至【代码】视图，添加 Index_Changed 事件，代码如下：

【示例代码】

```
Sub Index_Changed(ByVal Sender As Object, ByVal E As EventArgs)
    Dim theStr As String
    '获取头像所对应的文件名
    theStr = selface1.Items(selface.SelectedIndex).value
    dispimg.ImageUrl = "~/pic/touxiang/" & theStr
End Sub
```

在选择头像时，使用了两个下拉列表框控件，其中一个为可见状态，供用户选择，ID 为 Selface；另一个则为隐藏状态，ID 为 Selface1。两个控件绑定的数据是完全一样的。这

里，通过 Selface.SelectedIndex 属性来获取用户所选择的头像在下拉列表框控件中的序号，再通过 Selface1.Items(selface.SelectedIndex).value 属性来获取隐藏下拉列表框控件对应该序号的值，即相应图片的文件名。

(5) 有一个关键处需要注意。即使用了下拉列表框控件 Selface 的回发事件，其数据是通过数据集动态绑定的。而在 Dreamweaver 8 所提供的用户控件 DreamweaverCtrls 中，其页面绑定的回发属性 PostBackBind 的值默认为 True，即当页面回发时数据将会重新绑定。这样将导致下拉列表框控件 Selface 的值始终是第一项。因此必须在代码中，将属性 PostBackBind 的值改为 False 如图 7.56 所示。

```
</MM:DataSet>
<MM:PageBind runat="server" PostBackBind="False" />
```

图 7.56　修改 PostBackBind 属性

至此，页面中事件代码的添加工作大部分已经完成，只剩下最后一项，那就是【提交】按钮的事件代码的添加。虽然，在 Dreamweaver 8 中提供了插入记录的快捷途径，但那仅限于所插入的记录项的值均可从页面中的表单元素中获取。而在本页面中，对于用户所选择的留言表情，无法直接通过表单元素获取，因此只能通过传统的代码来实现。

(6) 为了与数据库连接，必须先导入相应的命名空间：System.Data 和 System.Data.OleDb，如图 7.57 所示。

```
<MM:PageBind runat="server" PostBackBind="False" />
<%@ Import Namespace="System.Data" %>
<%@ Import Namespace="System.Data.OleDb" %>   导入命名空间
<script runat="server">
```

图 7.57　导入命名空间

(7) 选择提交按钮控件并右击，从弹出的快捷菜单中选择【编辑标签】命令，在弹出的【标签编辑器-Buttou】对话框中选择【事件】|OnClick 分类，在右边的文本框中输入“sure_click”(即单击【提交】按钮时所触发的事件名称)，如图 7.58 所示。

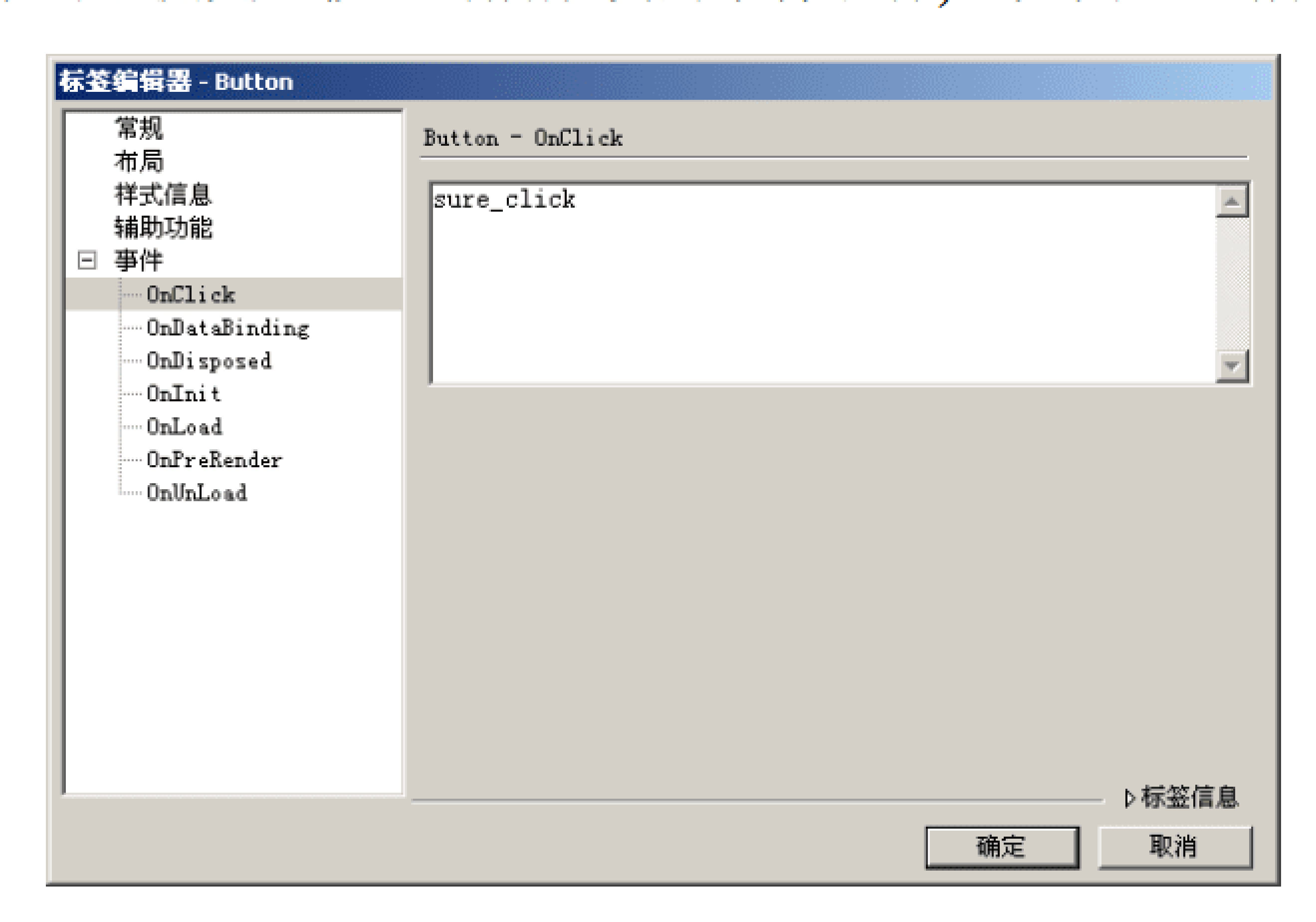


图 7.58　定义提交按钮的 OnClick 事件

(8) 单击【确定】按钮，将视图切换至【代码】视图，添加 sure_click 事件，代码如下：

【示例代码】

```
Sub sure_click(Sender As Object,E As Eventargs)
  '定义数据库连接
  Dim Cnn As OleDbConnection
  Dim Cmd As OleDbCommand
  Dim StrCnn As String
  Dim Sql As String
  Dim theTitle As String = title.Text  '获取留言标题信息
  Dim theLyr As String = lyr.Text  '获取留言人信息
  Dim theEmail As String = email.Text  '获取邮箱地址信息
  Dim theQQ As String = QQCode.Text  '获取 QQ 号码信息
  Dim theHomePage As String = homepage.Text  '获取个人主页信息
  Dim theContent As String = Content.Text  '获取留言内容信息
  Dim theSex As String
  Dim theQQH As String
  Dim theFace As String = selface.SelectedItem.Value '获取用户所选择的头像
  Dim theBQ As String = "0"
  '判断性别
  If Radio1.Checked Then
    theSex = "男"
  Else
    theSex = "女"
  End If
  '判断是否为悄悄话
  If qqh.Checked Then
    theQQH = "1"
  Else
    theQQH = "0"
  End If
'由于模板(ItemTemplate)的重复循环，在数据列表中的 CheckBox 控件具有同一 ID(即 bq)
'可以通过在 DataList 控件的 Items 集合中遍历并检索指定索引的 DataListItem，然后调用
相应的 FindControl 方法来查找 ID 名为 bq 的 CheckBox 控件，并判断当前是否处于选中状态
  Dim ObjCHK As CheckBox
  Dim i As Integer
  For i = 0 To DataList1.Items.Count - 1
    '查找 ID 为 bq 的 CheckBox 控件
    ObjCHK = DataList1.Items(i).FindControl("bq")
    '判断当前 CheckBox 控件是否处于选中状态
    If ObjCHK.Checked Then
      '获取选中的 CheckBox 控件的 ToolTip 属性，即该表情在数据库中的编号
      theBQ = ObjCHK.ToolTip
      '由于同一时间只可能有一个 CheckBox 控件处于选中状态，因此当检索到一个 CheckBox
      被选中，即可退出循环
      Exit For
    End If
  Next
  '连接数据库
  StrCnn =
System.Configuration.ConfigurationSettings.AppSettings("MM_CONNECTION_ST
RING_Cnn")
```

```
  Cnn = New OleDbConnection(StrCnn)
  Cnn.Open()
  '执行插入语句
  Sql = "insert into
lyinfo(name,sex,qq,email,homepage,title,content,faceid,biaoqingid,tag_qq
h) values('" & Trim(theLyr) & "','" & Trim(theSex) & "','" & Trim(theQQ) &
"','" & Trim(theEmail) & "','" & Trim(theHomePage) & "','" & Trim(theTitle)
& "','" & Trim(theContent) & "'," & Trim(theFace) & "," & Trim(theBQ) & ",'"
& Trim(theQQH) & "')"
  Cmd = New OleDbCommand(Sql, Cnn)
  Cmd.ExecuteNonQuery()
  Cnn.Close()
  '数据保存成功后，跳转至留言查询页面
  Response.Redirect("default.aspx")
End Sub
```

至此，本页面的制作全部完成，预览后的界面如图 7.59 所示。

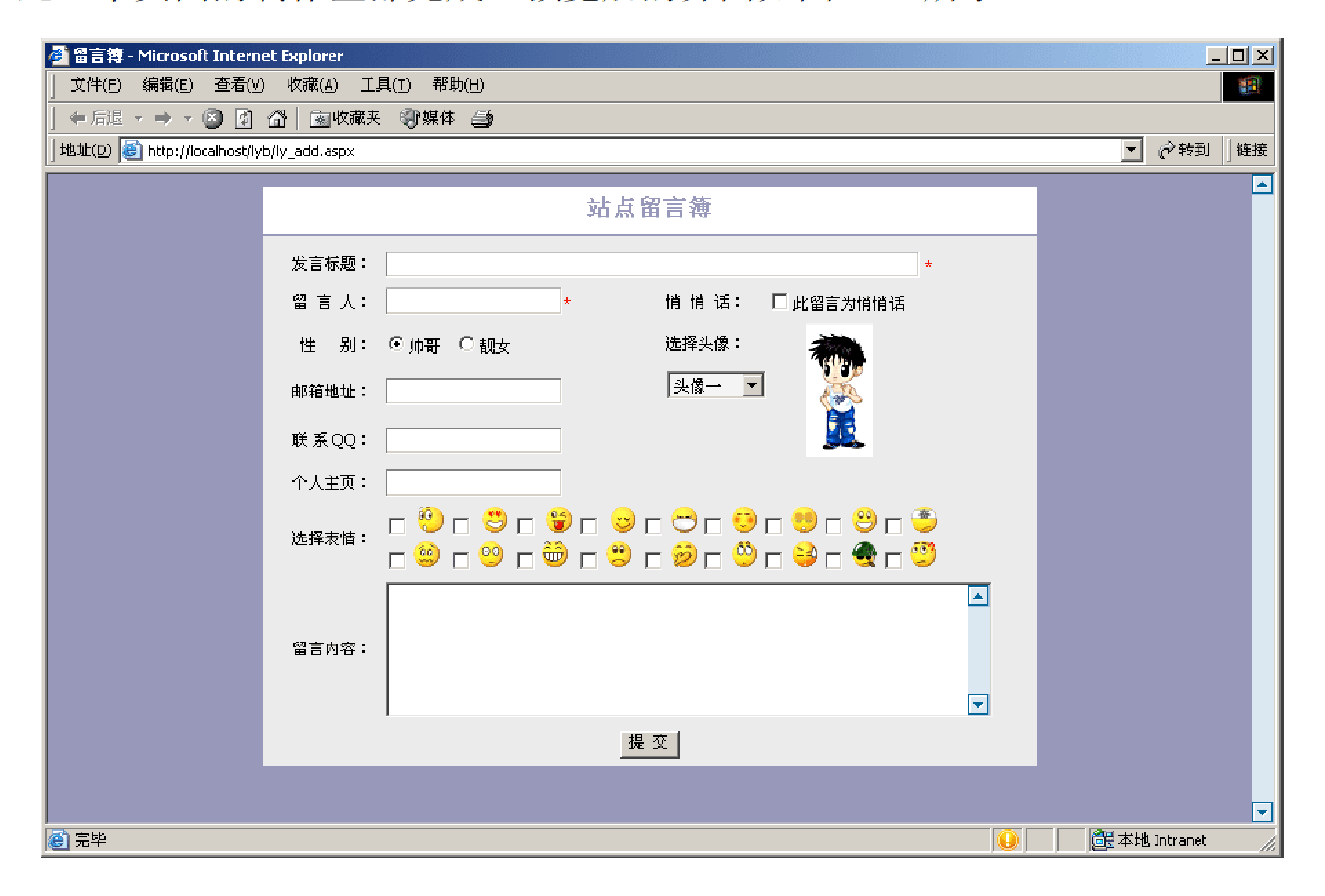


图 7.59　页面预览

7.3　留言簿的后台页面制作

留言簿的后台功能包括对留言的管理及留言回复。其中，留言管理功能又包括屏蔽留言和删除留言。

7.3.1　留言管理页面

留言管理页面与留言浏览页面类似，不同之处在于，在留言管理页面中无须对留言信息进行悄悄话和是否屏蔽的判断，同时添加对留言管理的操作按钮或链接，如回复、删除和屏蔽等。

1. 调整页面结构

(1) 将留言浏览页面复制并重命名为 Admin.aspx。在 Dreamweaver 8 中打开文件 Admin.aspx，将视图切换至【代码】视图，把用于判断留言是否为悄悄话及是否屏蔽的代码段删除，如图 7.60 所示。

```
<td height="78" valign="top">    
<%
  '获取当前留言信息的tag_hide字段值，该字段表示当前留言是否为被管理员设为屏蔽
  Dim strHide As String =dataset1.defaultView.Table.Rows(n_count)("tag_hide")
  '获取当前留言信息的tag_hide字段值，该字段表示当前留言是否为悄悄话
  Dim strQQH As String = dataset1.defaultView.Table.Rows(n_count)("tag_qqh")
  Dim str_HF As String = ""
  '判断当前留言是否有回复信息，如果有回复，则将其赋值给变量str_HF
  If Not IsDBNull(dataset1.defaultView.Table.Rows(n_count)("reply")) Then
    str_HF = dataset1.defaultView.Table.Rows(n_count)("reply")
  End If
  If strHide = "1" Then
'显示屏蔽信息
%>
  此留言已被管理员屏蔽，^_^
<% ElseIf strQQH = "1" Then '显示悄悄话信息 %>
  此留言为悄悄话，^_^
<% Else '显示留言内容 %>
  <%# dataset1.FieldValue("Content", Container) %>
<%
  End If
  If Not IsDBNull(str_HF) And str_HF <> "" Then
'如果存在回复信息，则显示相关回复
%>
  <br /><br />
```

删除

图 7.60　删除代码段

(2) 添加相应的管理操作图片，作为按钮。其中，回复操作仅是一个链接，即链接至相应的回复页面(该页面将在下一小节中叙述)，因此只需创建一个 Image 控件并设置相应链接即可，而删除和屏蔽操作均通过图片按钮控件(即 ImageButton 控件)来实现。

在个人头像下添加文本“管理：”，并将其字体设为粗体。在该文本后插入 pic 目录下的 Reply.gif(回复图片)，在【属性】面板中设置其边框属性为 0，链接属性为“LY_Reply.aspx?id=<%# DataSet1.FieldValue("ID", Container) %>”，这里将数据集 DataSet1 中的字段 ID 的值作为参数进行传递。选择图片并右击，在弹出快捷菜单中选择【编辑标签】命令。在接着弹出的【标签编辑器】对话框中将【常规】分类中的对齐属性设置为【绝对居中】。

(3) 在【ASP.NET-插入】工具栏中，拖动【图像】按钮至回复图片之后，此时将弹出【asp:图像按钮】对话框，如图 7.61 所示。

图 7.61　添加图像按钮 Del

(4) 在对话框中，设置 ID 为 Del，【图像 URL】选项为“pic/del.gif”，【工具提示】选项为“删除留言”，【命令名称】选项为 DelClick，【命令参数】选项为“<%#

DataSet1.FieldValue("ID", Container) %>”。单击【确定】按钮，完成插入操作，这里添加的是用于执行删除操作的命令按钮。选择该图像按钮，单击【属性】面板中的按钮，可弹出图像按钮的【标签编辑器】对话框，如图 7.62 所示。

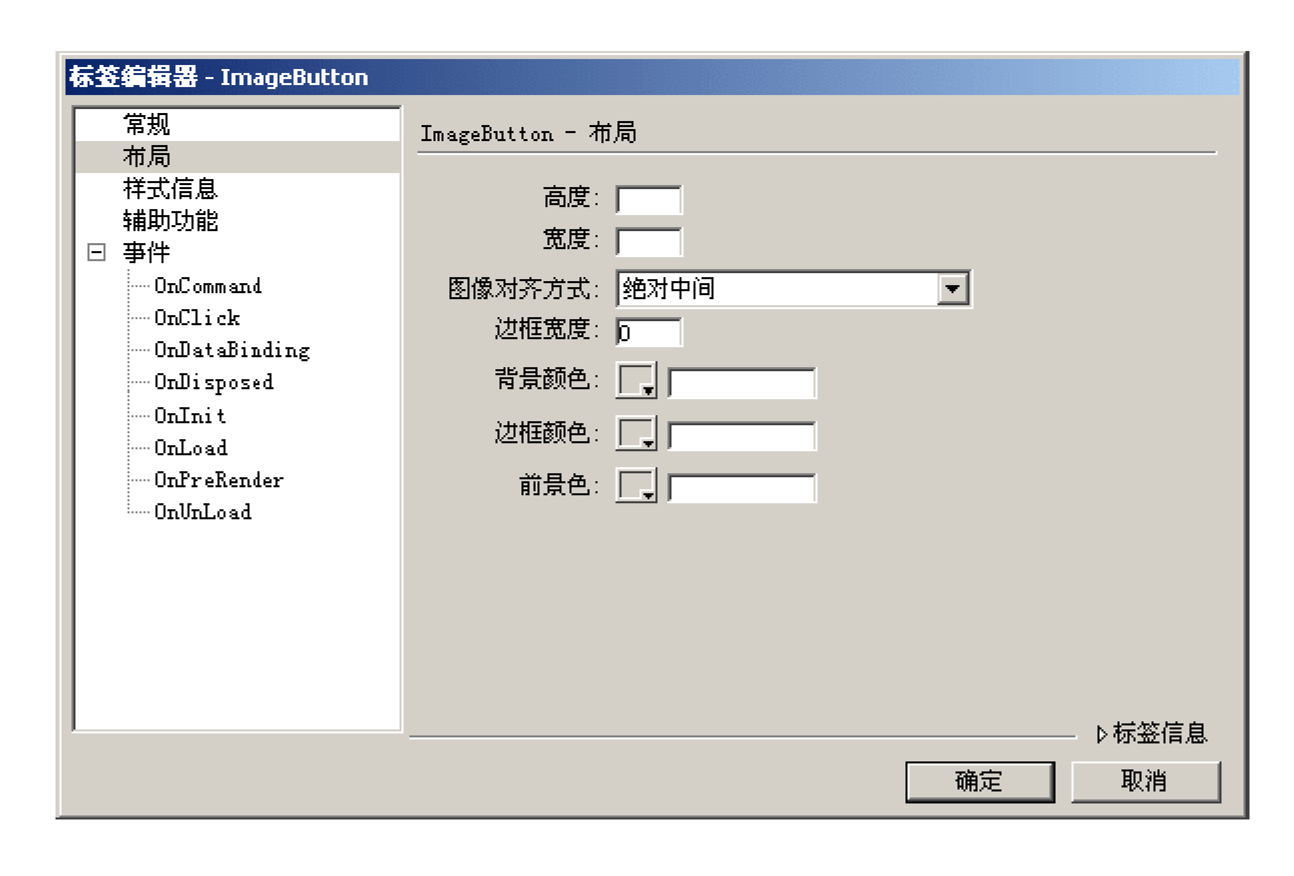


图 7.62　【标签编辑器-ImageButton】对话框

(5) 选择【布局】分类，设置【图像对齐方式】选项为【绝对中间】，边框宽度为 0。单击【确定】按钮，完成设置。

(6) 与以上操作类似，在图像按钮 Del 后添加一个新的用于执行屏蔽操作的图像按钮，其属性设置如图 7.63 所示。

图 7.63　添加图像按钮 Hide

至此，页面结构调整完毕。接下来，是编写相应操作的事件代码。

2. 编写相应事件代码

(1) 选择 Repeater 数据控件并右击，在弹出的快捷菜单中选择【编辑标签】命令，此时将弹出针对 Repeater 控件的标签编辑器，如图 7.64 所示。

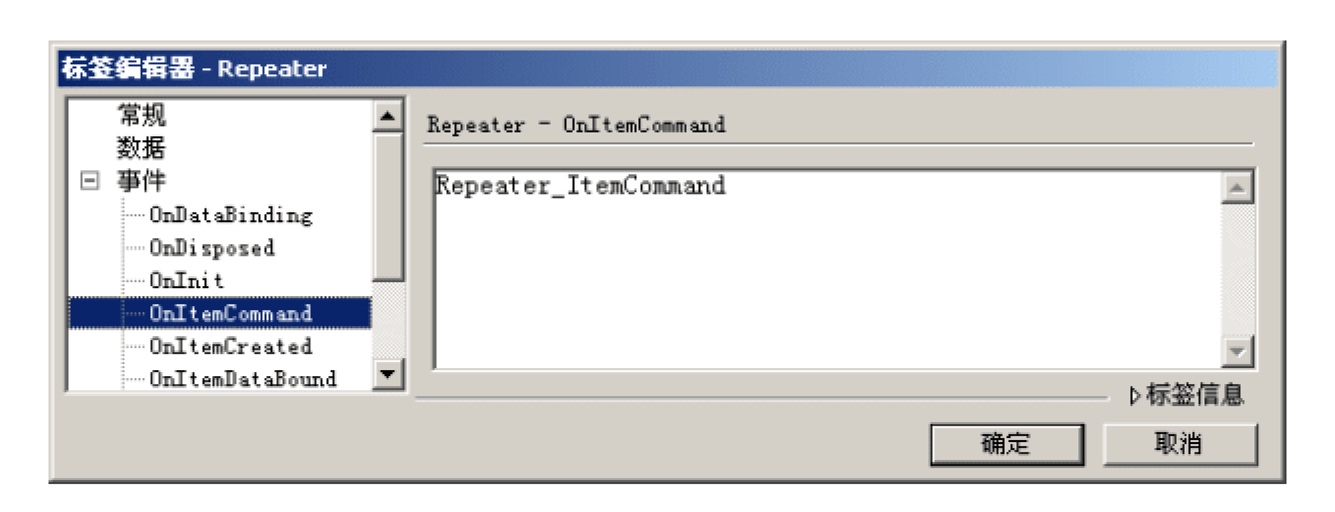


图 7.64　【标签编辑器-Repeater】对话框

(2) 在【事件】分类中选择 OnItemCommand，在右侧的文本框中输入定义的事件名称 Repeater_ItemCommand。单击【确定】按钮，将视图切换至【代码】视图，添加 Repeater_ItemCommand 事件，代码如下：

【示例代码】

```
Sub Repeater_ItemCommand(ByVal source As Object, ByVal e As
RepeaterCommandEventArgs)
  '定义数据库连接
  Dim Cnn As OleDbConnection
  Dim Cmd As OleDbCommand
  Dim StrConn As String
  Dim Sql As String
  Dim theID As String
  '获取数据库连接字符串
  StrConn =
System.Configuration.ConfigurationSettings.AppSettings("MM_CONNECTION_ST
RING_Cnn")
  '通过 e.CommandName 获取当前触发事件的命令名，以此判断是屏蔽操作还是删除操作
  If e.CommandName = "HideClick" Then
    '命令名为 HideClick，当前操作为屏蔽留言操作
    theID = e.CommandArgument  '获取命令参数，该参数表示当前所要操作的留言 ID
    '连接数据库
    Cnn = New OleDbConnection(strConn)
    Cnn.Open()
    '将当前留言的 tag_hide 字段值设为 1，即置为屏蔽状态
    Sql = "update lyinfo set tag_hide='1' where id=" & Trim(theID)
    Cmd = New OleDbCommand(Sql, Cnn)
    '执行 SQL 语句
    Cmd.ExecuteNonQuery()
    Cnn.Close()
  ElseIf e.CommandName = "DelClick" Then
    '命令名为 DelClick，当前操作为删除留言操作
    theID = e.CommandArgument  '获取命令参数，该参数表示当前所要操作的留言 ID
    Cnn = New OleDbConnection(strConn)
    Cnn.Open()
    '执行删除语句
    Sql = "delete from lyinfo where id=" & Trim(theID)
    Cmd = New OleDbCommand(Sql, Cnn)
    Cmd.ExecuteNonQuery()
    Cnn.Close()
    '将页面重定向本页面，刷新数据显示
     Response.Redirect("admin.aspx")
  End If
End Sub
```

这里，采用了 DataList 控件的 ItemCommand 事件，而没有定义图像按钮的单击事件。这是由于 ItemCommand 事件是 DataList 控件自带的默认事件，任何 DataList 控件中的命令按钮被单击后，首先触发的是 ItemCommand 事件，然后才是用户定义的相应的按钮事件。

(3) 如果当前系统是运行在 ASP.NET 2.0 的环境下，将可能出现运行时异常错误：

Invalid postback or callback argument(无效的回发或回调参数)。在此情况下，需要在页首添加“EnableEventValidation="false"”，如图 7.65 所示。

```
<%@ Page Language="VB" ResponseEncoding="gb2312" EnableEventValidation="false"%>
<%@ Register TagPrefix="MM" Namespace="DreamweaverCtrls" Assembly=
"DreamweaverCtrls,version=1.0.0.0,publicKeyToken=836f606ede05d46a,culture=neutral" %>
```

图 7.65　修改代码

事实上，在 ASP.NET 1.0 中并没有 EnableEventValidation 属性，该属性是 ASP.NET 2.0 中新增的，用于指示是否执行事件验证，默认值为 True。事件验证机制在一定程度上可消除未经授权的回发请求和回调所带来的风险，它指示 ASP.NET 仅对会在回发请求或回调期间在控件上所引发的特定事件进行验证。这里，为了保证事件的正常运行，将其进行屏蔽。

至此，本页面功能全部实现，预览界面如图 7.66 所示。

图 7.66　页面预览

7.3.2　留言回复页面

在上一小节中，介绍了留言管理页面的实现。其中，为留言的回复创建了一个链接，链接的页面为 LY_Reply.aspx。下面，将介绍该页面的具体实现。

1. 页面的基本布局

按如图 7.67 所示布局页面。

在此页面中，根据页面传递的参数 ID，获取所要回复的留言的相关信息，包括标题和内容。页面中，所需输入的内容仅仅包括回复内容。

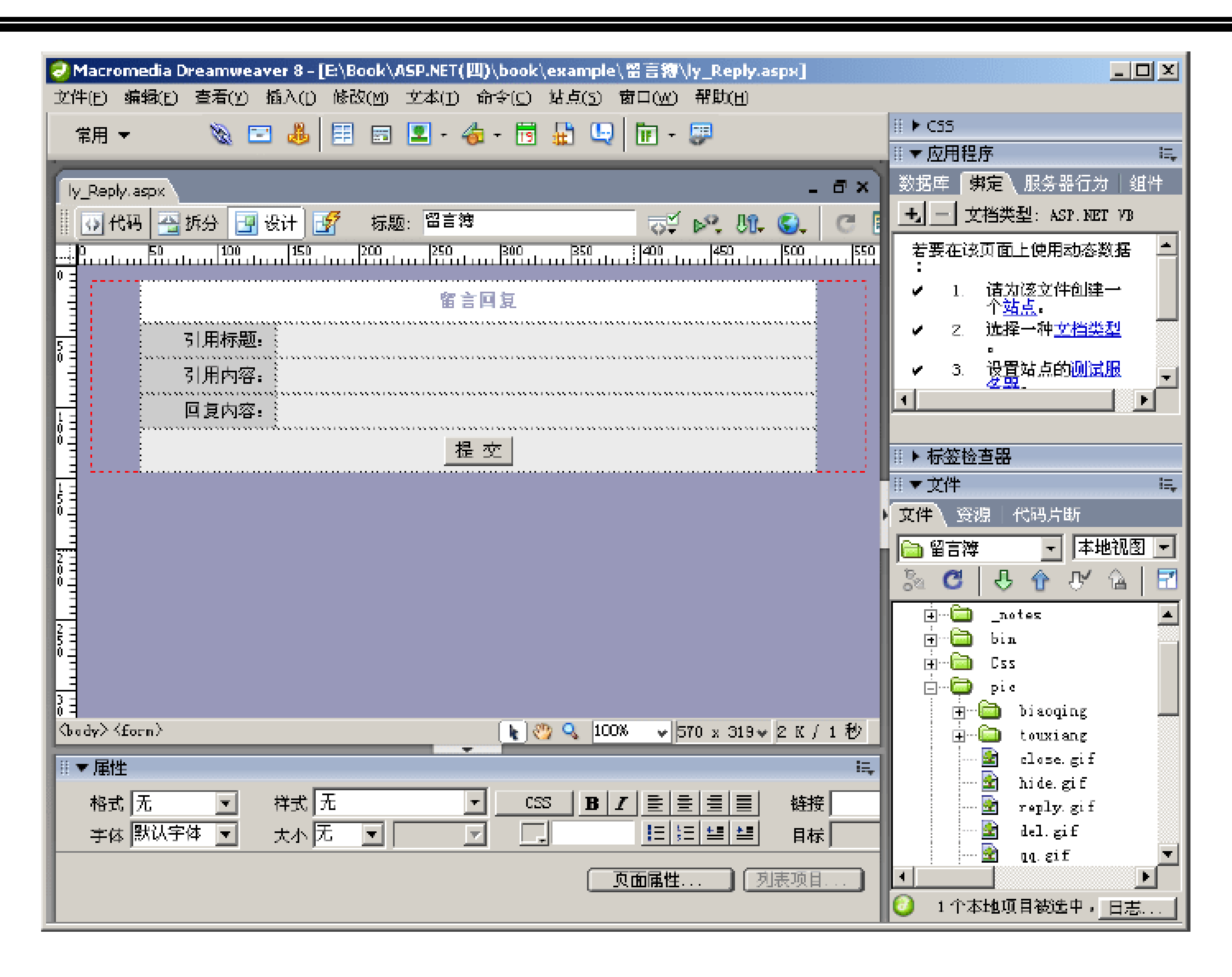


图 7.67　页面基本框架

2. 添加相应的 ASP.NET 控件

(1) 先添加一个数据集，用于获取指定 ID 的留言信息。

打开【应用程序】面板组，切换至【服务器行为】面板，单击 按钮，在弹出的菜单中选择【数据集】命令，将弹出【数据集】对话框，如图 7.68 所示。

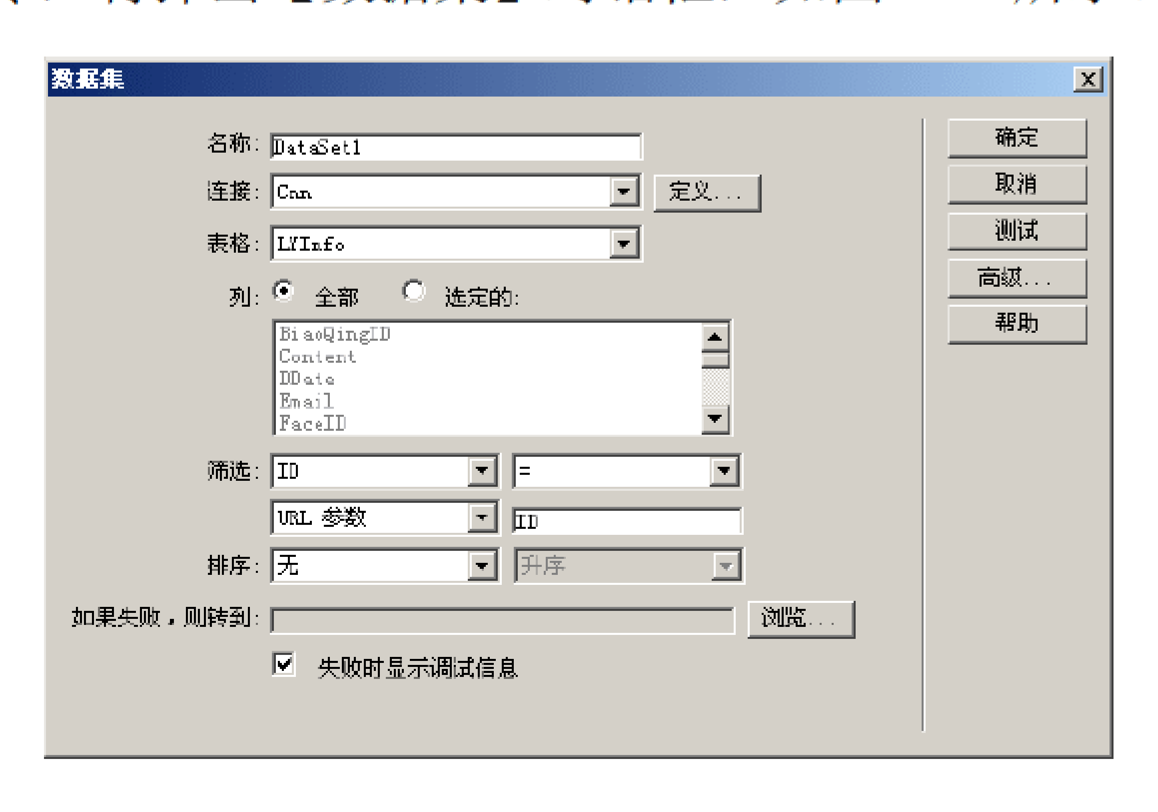


图 7.68　【数据集】对话框

(2) 在此对话框中，设置数据集名称为 DataSet1，连接为 Cnn，选择数据表 LYInfo，同时设置筛选条件为字段 ID 等于 URL 参数 ID。

(3) 在【ASP.NET-插入】工具栏中，拖动标签控件至表格中文本“引用标题”后的单元格中，此时将弹出【asp:标签】对话框，如图 7.69 所示。

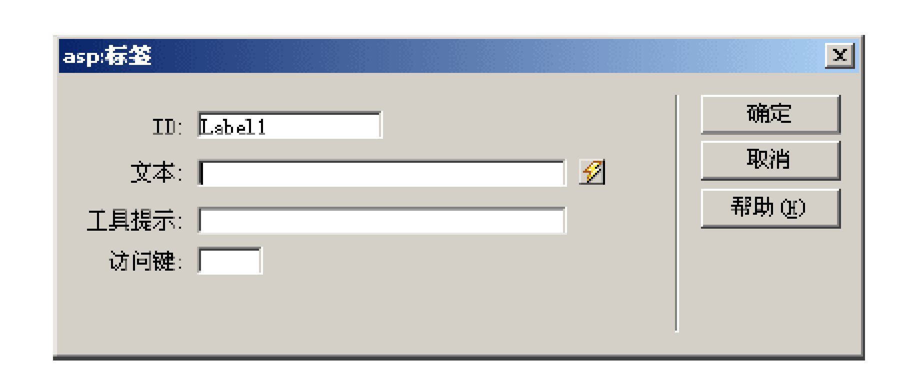


图 7.69　设置标签控件

单击【绑定动态数据】按钮，在弹出的【动态数据】对话框中，选择数据集 DataSet1 下的字段 Title，如图 7.70 所示。

图 7.70　【动态数据】对话框

单击【确定】按钮，返回【asp:标签】对话框，再次单击【确定】按钮，完成该标签的设置。

(4) 重复以上操作，添加新的标签控件至引用内容后的单元格中。将标签名设置为 Label2，同时单击【绑定动态数据】按钮，选择数据集 DataSet1 下的字段 Content 作为其显示文本。

(5) 在【ASP.NET-插入】工具栏中，拖动文本框控件至回复内容后的单元格中，此时将弹出【asp:文本框】对话框，如图 7.71 所示。

设置其 ID 为 Reply，文本模式为多行，列数为 44，行数为 6，同时单击【绑定动态数据】按钮，选择数据集 DataSet1 下的字段 Reply 作为其文本内容。

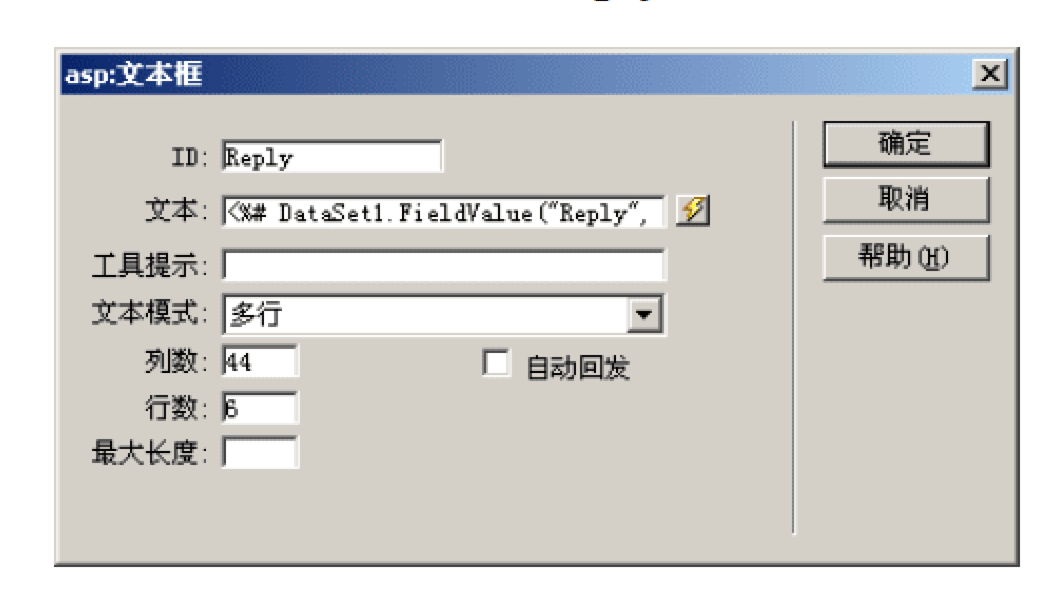


图 7.71　添加文本框控件 Reply

(6) 为了便于更新记录，还需添加一个隐藏域来存储指定要更新记录的 ID，该 ID 的值可从页面的 URL 参数获取，也可从数据集中的字段 ID 获取。

单击【ASP.NET-插入】工具栏中的【更多标签】按钮，在弹出的【标签选择器】对话框的【HTML 标签】|【表单】分类中，选择“input type="Hidden"”，如图 7.72 所示。

单击【插入】按钮，在弹出的【标签编辑器】对话框中，设置其名称为“lyid”，值为“<%# DataSet1.FieldValue("id", Container) %>”，如图 7.73 所示。

单击【确定】按钮，完成隐藏域的添加操作。

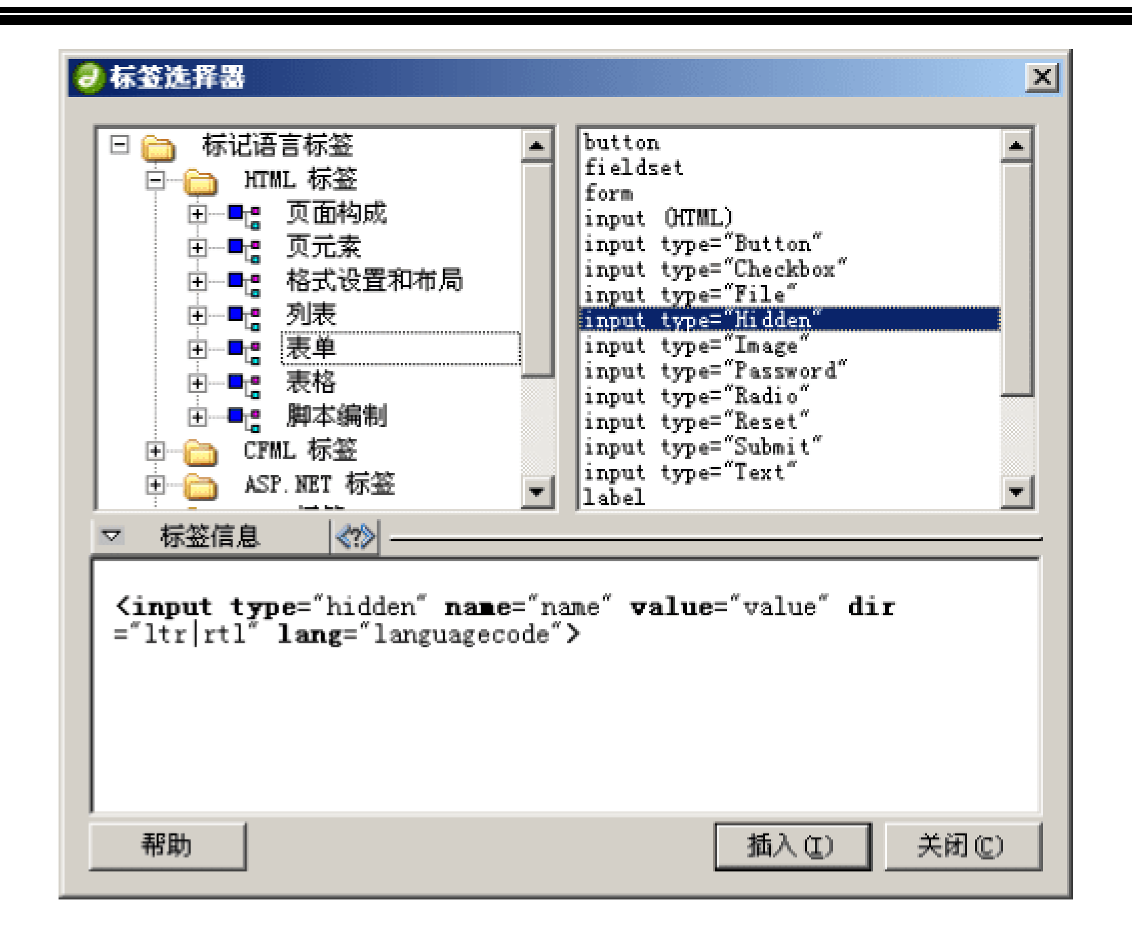


图 7.72　添加隐藏域

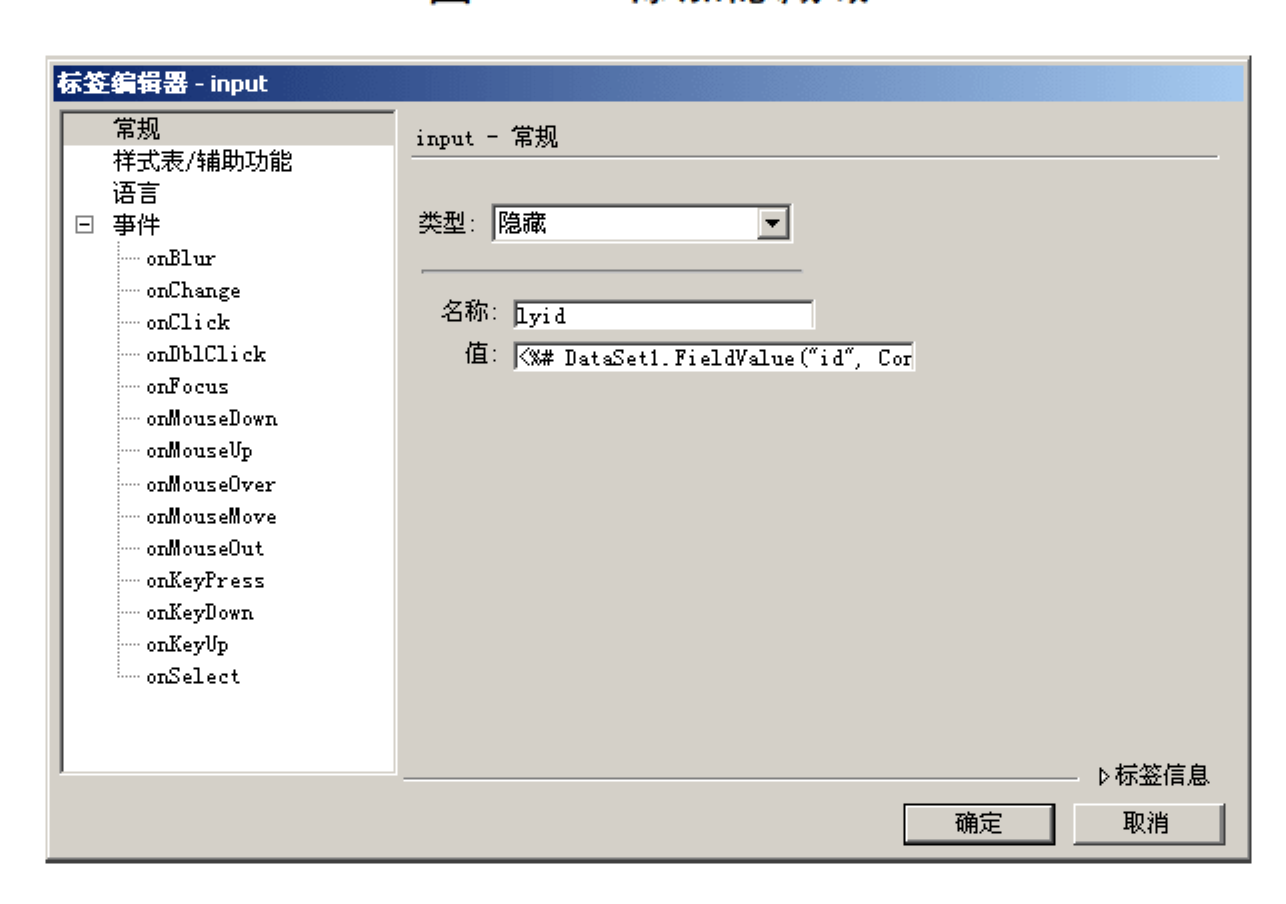


图 7.73　隐藏域的标签编辑器

3. 添加更新记录功能

打开【应用程序】面板组，切换至【服务器行为】面板，单击![+]按钮，在弹出的菜单中选择【更新记录】命令，将弹出【更新记录】对话框，如图 7.74 所示。

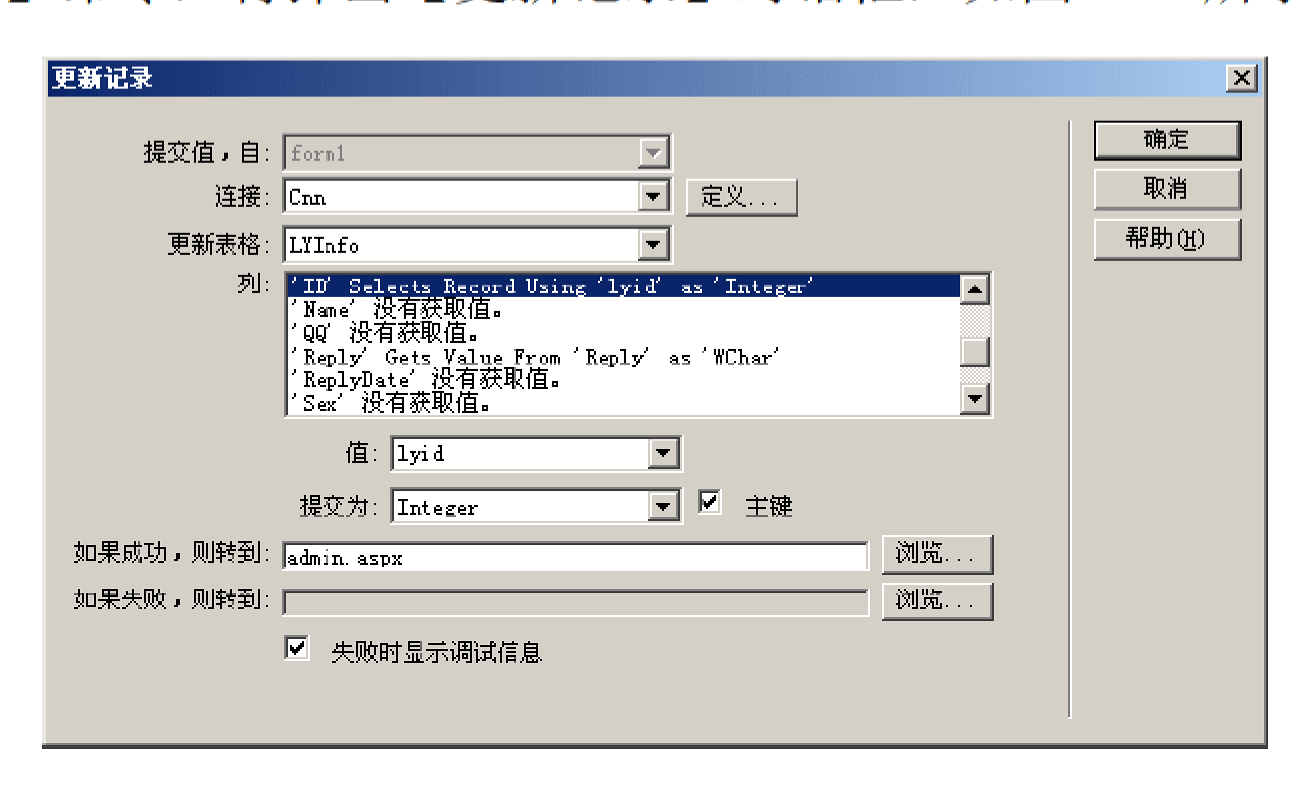


图 7.74　【更新记录】对话框

在【提交值，自】下拉列表框中选择当前页面中的表单元素 Form1，在【连接】下拉列表框中选择 Cnn，在【更新表格】下拉列表框中选择 LYInfo。在【列】列表框中需要进

行操作的列包括 ID 和 Reply 两列。选择 ID 列，在【值】下拉列表框中选择前面定义的隐藏域标识 lyid；选择 Reply 列，在【值】下拉列表框中选择前面定义的文本框控件 Reply。最后，在【如果成功，则转到】文本框中输入“admin.aspx”，即当数据更新成功后页面自动跳转至 admin.aspx 页面(留言管理页面)。单击【确定】按钮，完成更新记录的设置。

至此，留言回复页面的功能全部完成，预览界面如图 7.75 所示。

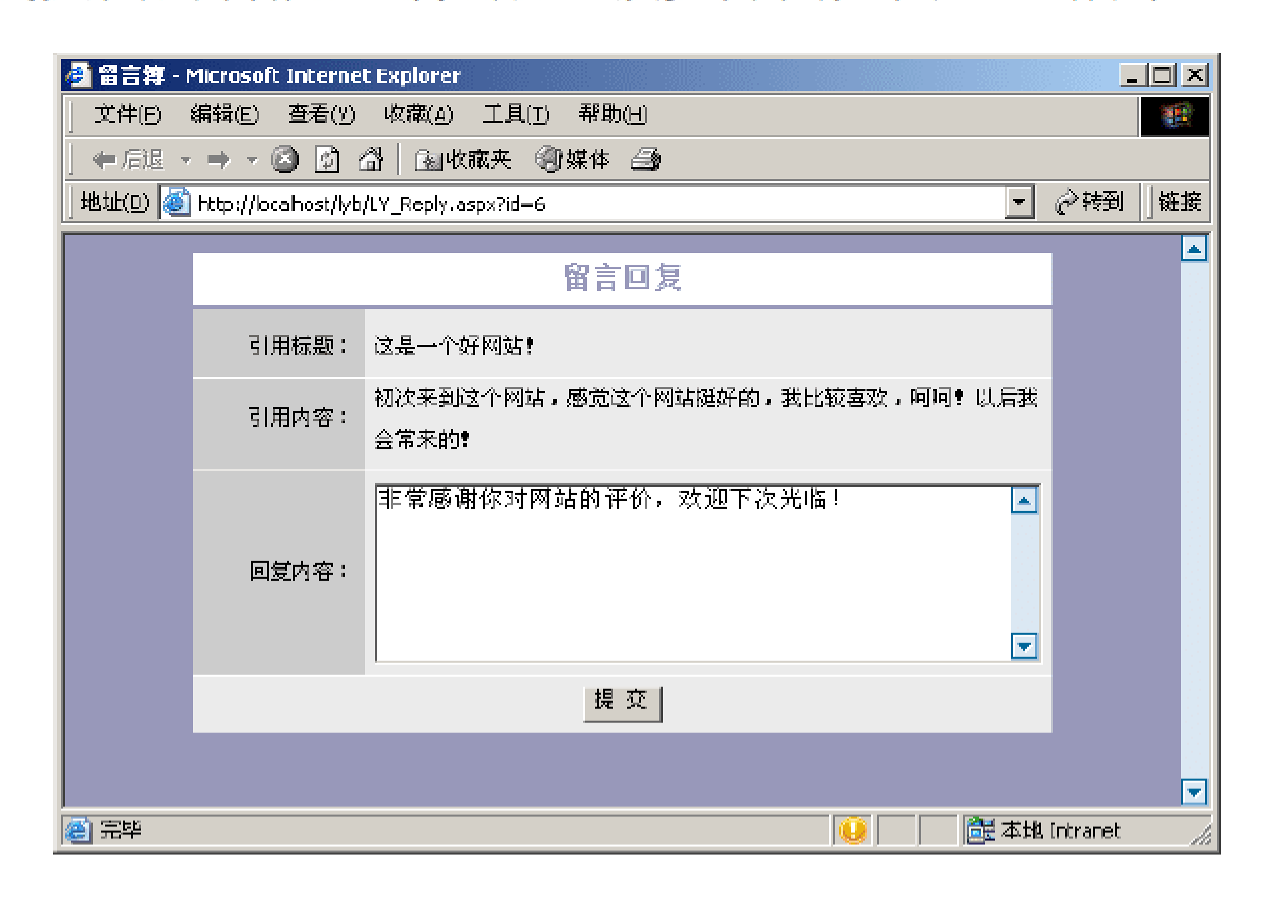


图 7.75　页面预览

相比之下，可以发现在 Dreamweaver 8 中实现记录的更新比通过编写代码实现更新记录要方便和快捷得多。

7.4　管理员登录页面

在该留言簿系统中，还有一个重要页面没有介绍，那就是管理员的登录页面。这个页面是非常重要的，对于普通的用户只能在前台的页面中进行操作，如添加留言、浏览留言等。而后台页面中的相关操作，仅在用户以管理员的身份登录后方可进行。

这一节将叙述管理员登录页面 Login.aspx 的实现。具体步骤如下。

(1) 页面基本框架的设计，如图 7.76 所示。

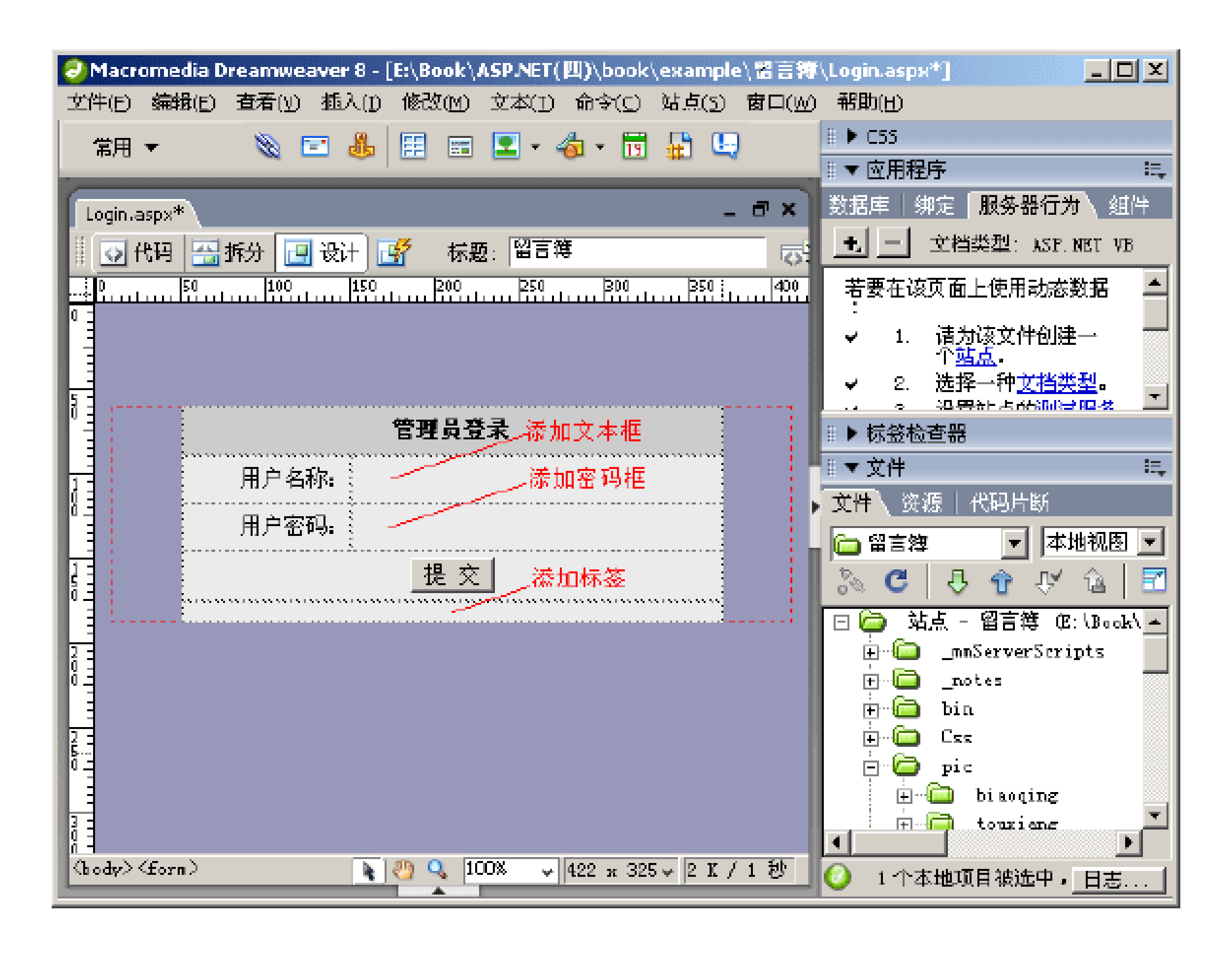


图 7.76　页面基本框架

可以看出，所添加的文本框用于输入管理员的用户名称，密码框用于输入用户密码，而标签则用于显示相关的错误信息。

(2)　添加用户名称对应的文本框控件。在【ASP.NET-插入】工具栏中，拖动文本框控件至用户名称后的单元格中，按如图 7.77 所示设置其属性。

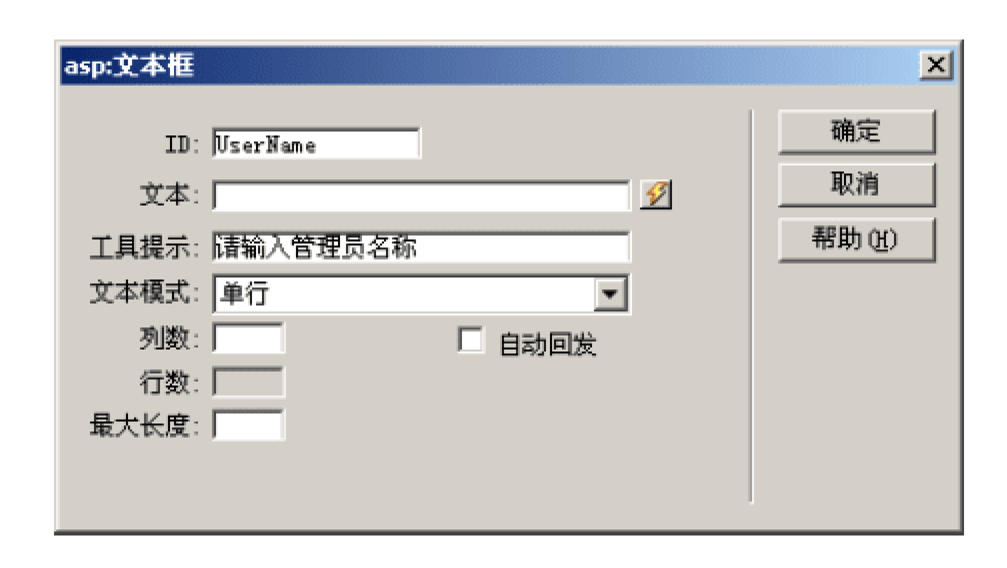


图 7.77　添加文本控件 UserName

(3)　添加用户密码对应的密码框控件。在【ASP.NET-插入】工具栏中，拖动文本框控件至用户密码后的单元格中，按如图 7.78 所示设置其属性。

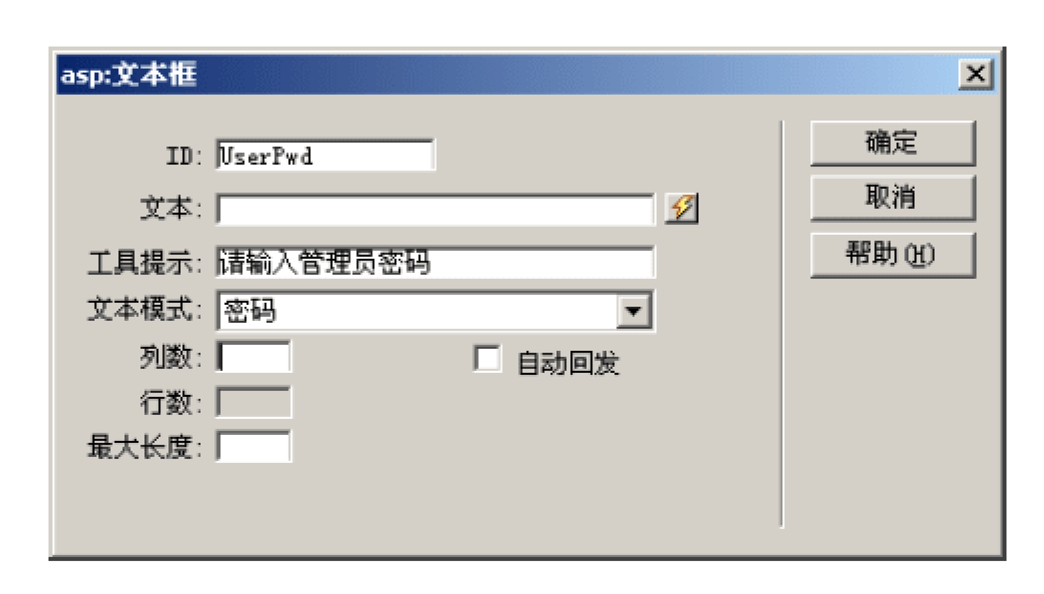


图 7.78　添加密码框控件 UserPwd

(4)　拖动【ASP.NET-插入】工具栏中的标签控件至【提交】按钮下的单元格中，设置其可见属性为不可见，如图 7.79 所示。

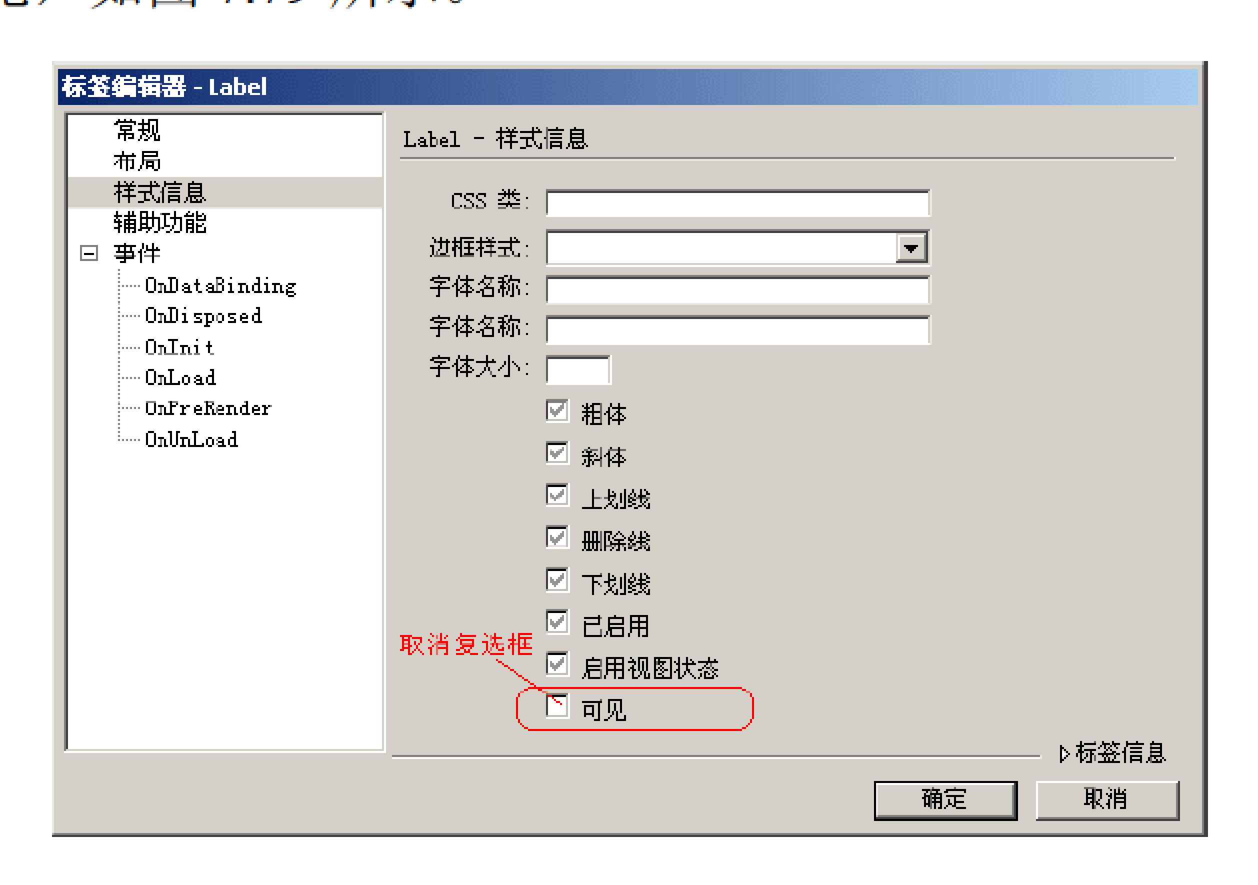


图 7.79　设置 Label 控件的可见属性

同时设置该标签控件的文本为“用户名及密码错误！”，设置其字体颜色为红色。

(5)　为了验证用户输入的正确性，需要创建一个数据集。打开【应用程序】面板组，切换至【服务器行为】面板，单击 ± 按钮，在弹出的菜单中选择【数据集】命令，将弹出

【数据集】对话框，如图 7.80 所示。

图 7.80 【数据集】对话框

(6) 设置数据集名称为 DataSet1，数据库连接选择 Cnn，在【表格】下拉列表框中选择 UserInfo，同时设置筛选条件：字段 UserName 等于表单变量 UserName 即数据表 UserInfo 中的字段 UserName 的值等于用户在表单元素 UserName 输入的值。但是，一个条件是不够的。判断一个用户是否为管理员，除了验证用户名还需验证所输入的密码是否正确。

(7) 单击【高级】按钮，在弹出的对话框中显示了包含过滤条件的 SQL 语句以及定义的参数，如图 7.81 所示。

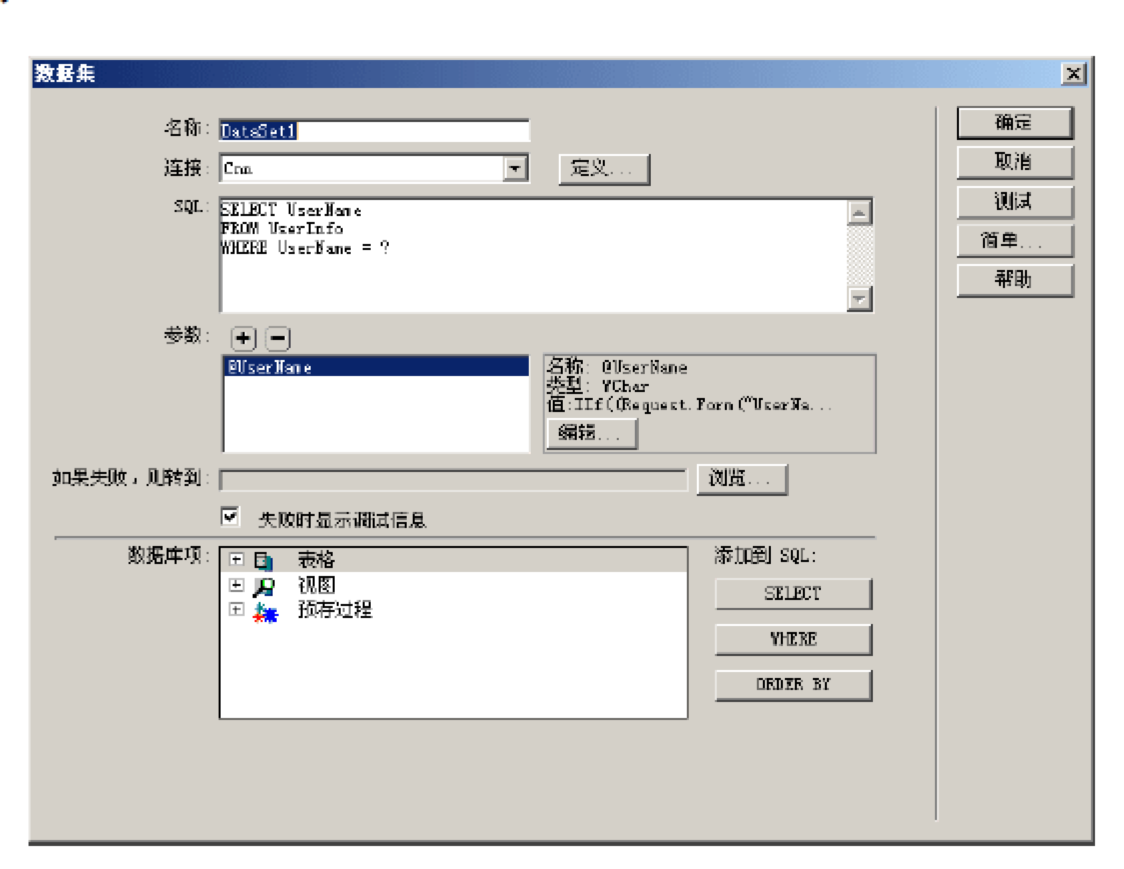


图 7.81 记录集的高级设置

(8) 单击按钮，添加一个新的参数。此时，将弹出【添加参数】对话框，如图 7.82 所示。输入参数名称“@UserPwd”，设置参数类型为 VarChar，参数的值可通过单击 Build 按钮打开【生成值】对话框来创建，如图 7.83 所示。

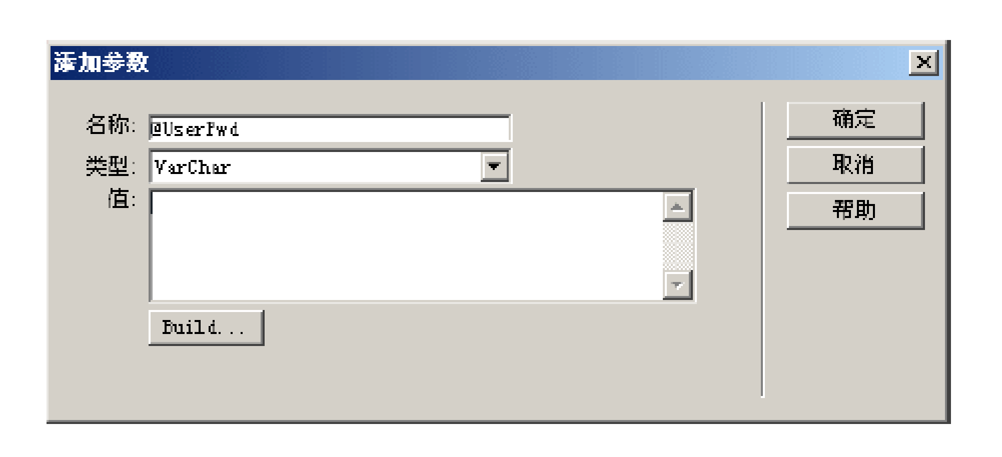


图 7.82 【添加参数】对话框

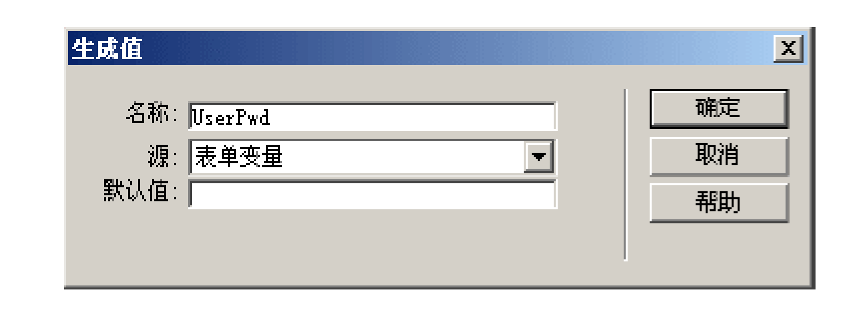


图 7.83 【生成值】对话框

在【名称】文本框中输入前面在表单中创建的密码框的 ID 值 UserPwd，从【源】下拉

列表框中选择【表单变量】，单击【确定】按钮。此时，在【添加参数】对话框中可以看到所创建的参数的值，如图 7.84 所示。

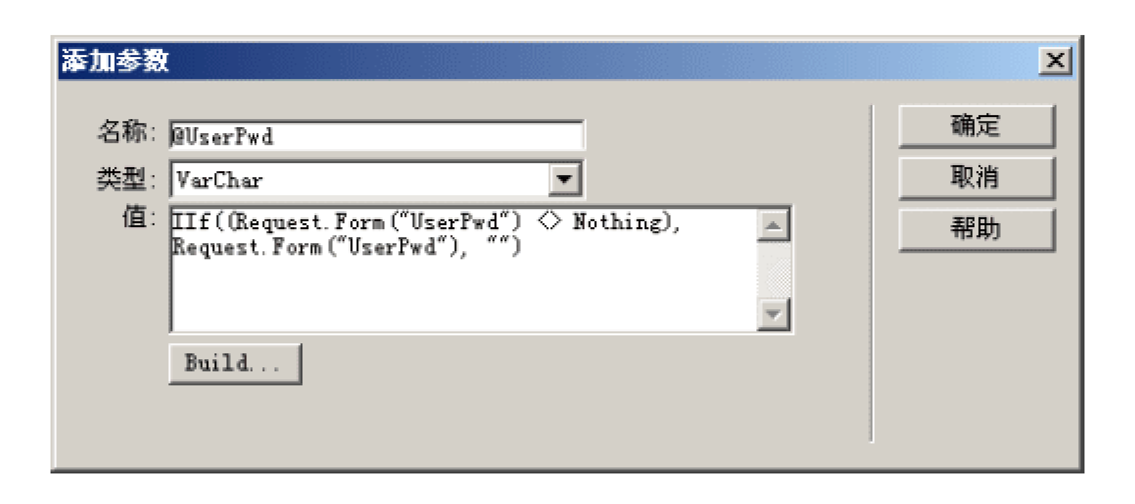


图 7.84 【添加参数】对话框

(9) 在参数@UserPwd 的值中，IIF 函数用于对表单变量 UserPwd 的值是否为 Null 进行判断。如果为空，则返回空字符串；如果不为空，则返回该表单变量的值。单击【确定】按钮，即可完成参数的添加。此时，还需在 SQL 语句中添加一个条件表达式以判断密码是否正确，如图 7.85 所示。

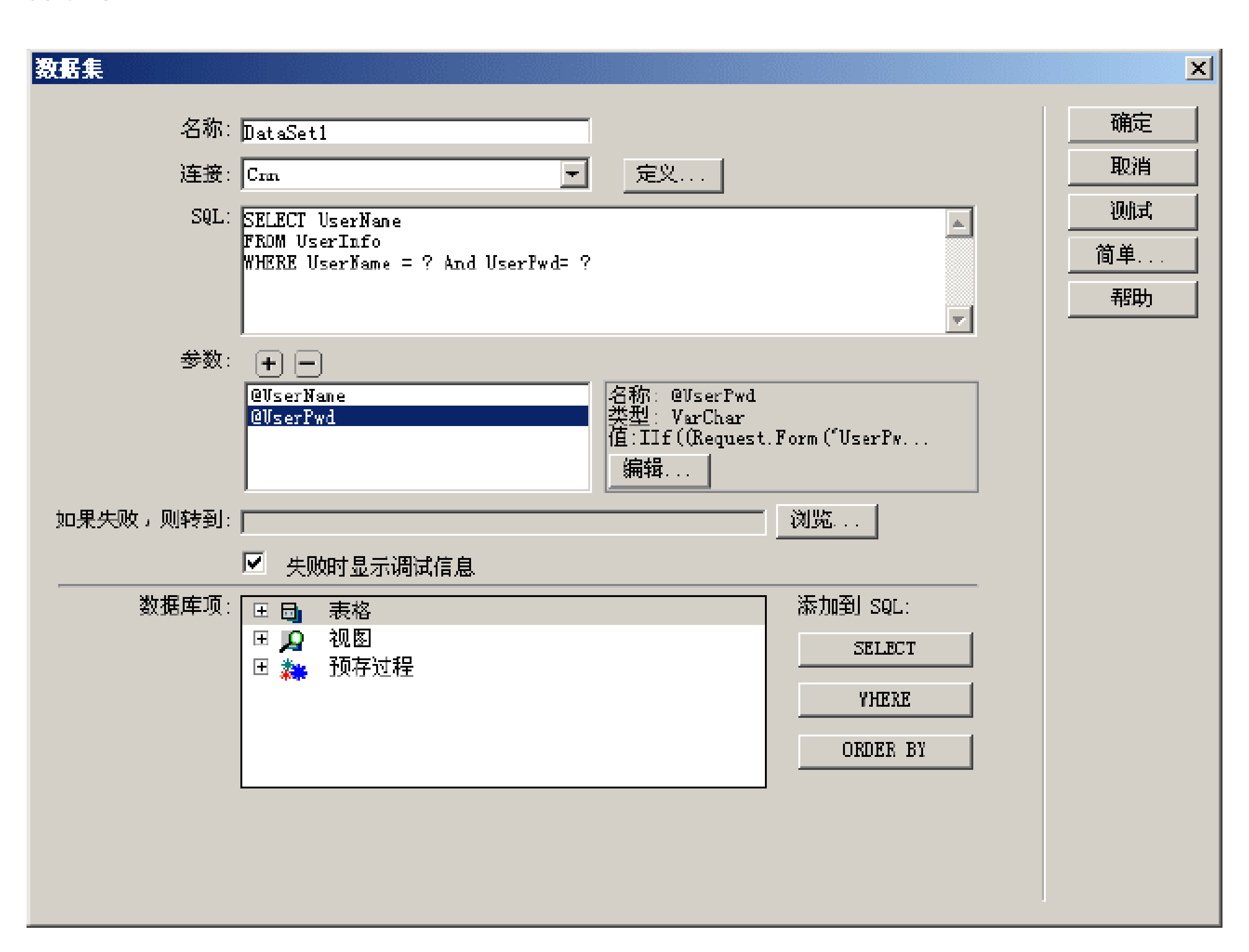


图 7.85 数据集的高级设置

(10) 单击【确定】按钮，完成数据集的创建。根据数据集中的记录是否为空，可以判断用户输入的用户名及密码是否正确。那么如何判断数据集中是否有记录呢，这可以通过绑定字段来实现。

打开【应用程序】面板组，切换至【绑定】面板，将数据集下的字段 UserName 拖至页面的任何位置，因为这里只用它来检查数据集中是否有记录，同时判断该绑定字段的值是否为空。如果不为空，则说明用户输入的用户名及密码正确，直接将页面跳转至留言管理页面 Admin.aspx，此操作可通过一个自定义的函数 CheckValue 来实现。

(11) 将视图切换至【代码】视图，找到绑定字段相应的代码，将其修改成调用自定义函数 CheckValue，如图 7.86 所示。

```
<tr>
  <td colspan=2 align=center><asp:Label ForeColor="#FF0000" ID="Label1" runat
="server" Text="用户名或密码错误！" Visible="false" />
  <%# CheckValue(DataSet1.FieldValue("UserName", Container)) %></td>
</tr>
```

图 7.86　修改代码

然后在代码中添加自定义函数 CheckValue 的代码以及【提交】按钮的单击事件，代码如下：

【示例代码】

```
<script Language="VB" runat="server">
 '定义函数 CheckValue
 Function CheckValue(Byval Str1 As String) As String
   if Str1<>nothing then
    '当所传递的绑定字段的值不为空时，验证成功，页面跳转至 Admin.aspx
    Response.redirect ("Admin.aspx")
   end if
 End Function
 Sub Sure_Click(ByVal Sender As Object, ByVal E As EventArgs)
   '显示错误信息
   Label1.visible=true
 End Sub
</script>
```

在【提交】按钮的单击事件 Sure_Click 中，只添加了一条语句，将标签 Label1 的 Visible 属性设置为 True，显示错误信息。这是由于如果执行到此事件代码，则只有一个原因，那就是用户输入的用户名或密码不正确。因为，如果验证成功，则页面在执行自定义函数 CheckValue 时便已跳转至留言管理页面了，而执行不到【提交】按钮的单击事件这一步。

至此，管理员登录页面的功能全部完成，预览界面如图 7.87 所示。

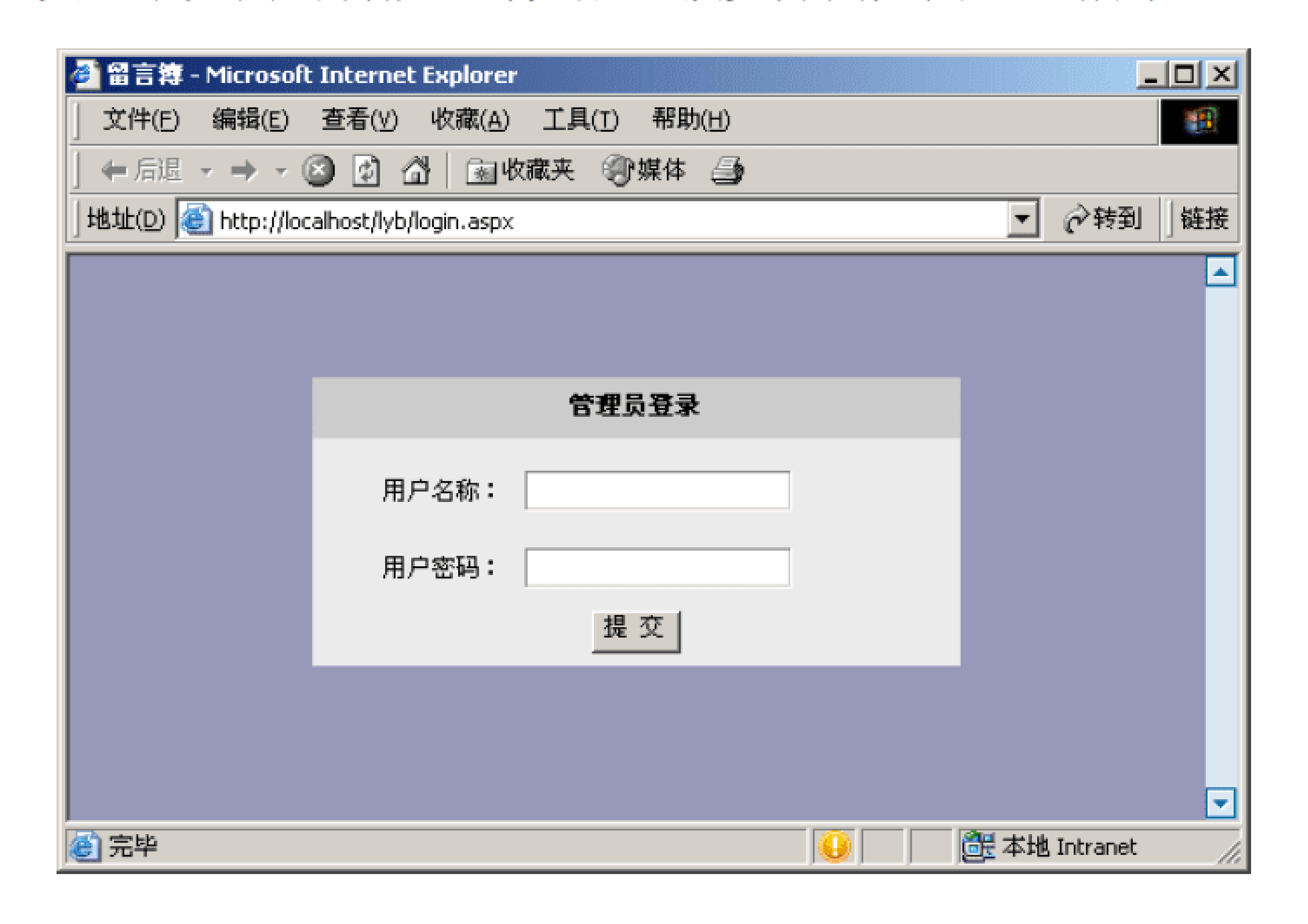


图 7.87　页面预览

7.5　习　　题

(1) 熟悉在 Dreamweaver 8 中操作各种 ASP.NET 控件的方法，掌握 ASP.NET 控件的添加和相关属性的设置。

(2) 添加用来新增和修改管理员信息的(允许修改用户名和密码)页面。

第 8 章　会 员 管 理

上一章介绍了留言簿的制作，相信读者对在 Dreamweaver 8 中开发 ASP.NET 应用程序已经有了一个基本的了解。这一章，将继续介绍会员管理系统的实现。

8.1　系 统 分 析

会员管理主要用于对站点的会员进行管理，它可以是一个独立的系统，也可被整合到其他的应用程序中，例如论坛。对于一个专业的网站，会员管理是必要的。这样不但可以为注册会员提供更多更好的服务，同时还也可以防止站内资源被其他网站非法盗链。

8.1.1　系统功能

作为一个最基本的会员管理系统，主要包括会员注册、会员登录、找回密码、修改信息等几项功能。而更为复杂的会员管理，还包括会员级别的设置、会员点数的设置以及浏览权限设置等多项功能，这里仅介绍会员管理系统中最基本的功能。

1. 会员注册

允许用户注册为站点的新会员。在注册的同时，要求用户输入正确的邮箱地址。在注册成功后，系统自动发送验证邮件至指定的邮箱中，用户收取邮件后，单击邮件中的链接，进行账户激活。只有激活，用户方可正常登录。

2. 会员登录

用户根据注册的用户名和密码进行登录。在进行验证时，不仅需要验证用户名和密码的正确性，还要判断用户是否已经激活账户。

3. 找回密码

当用户忘记自己的密码时，可通过此操作来找回自己的密码。在操作中，用户需提供用户名以及正确的出生日期，并回答在注册时所设置的密码提示问题的答案。

4. 修改信息

修改用户注册时所填写的个人信息，同时提供修改密码的功能。

8.1.2　数据库的建立

本系统使用 Microsoft Access 2000 数据库，其数据库名为 UserInfo.Mdb，在该数据库中仅有 UserSheet(用户信息表)一个数据表。

1. 用户信息表(UserSheet)

UserSheet 数据表主要用于存储用户的基本信息，其表结构见表 8.1。

表 8.1　UserlSheet 数据表

字 段 名	数据类型	字段描述	字段大小	默 认 值	主　键
Code	自动编号	用户编号	长整型		
LoginName	文本	表示用户的登录名称	50		P
UserPass	文本	表示用户的登录密码	50		
UserName	文本	表示用户的真实名称	50		
Birth	日期/时间	表示出生日期			
Email	文本	表示邮箱地址	100		
PassQuestion	文本	表示密码提示问题	100		
PassAnswer	文本	表示密码提示问题的答案	100		
Address	文本	表示联系地址	100		
Tel1	文本	表示联系的固定电话	50		
Tel2	文本	表示联系的移动电话	50		
Works	文本	表示用户的工作单位	100		
ZipCode	文本	表示邮政编码	50		
Sex	文本	表示用户性别	50		
JHTag	文本	账号是否激活标识	1	0	

说明：字段 JHTag 表示用户的账号是否激活；其值为 0 表示未激活，为 1 表示已激活，默认值为 0。

8.1.3　站点设置

本例中，站点设置的本地信息如图 8.1 所示。

其中，站点名称设置为“会员管理”，本地根文件夹设置为会员管理系统所在的磁盘目录，默认图像所在的文件夹设置为会员管理系统下的 Image 目录。此外，HTTP 地址为“http://localhost/user”。当然，在此之前需用要将会员管理系统设置为 Web 共享，其共享名为“user”。

站点设置中，远程信息设置如图 8.2 所示。

由于对用户管理系统的设计、测试和运行均是在本机上操作的，因此这里将访问方式设置为【本地/网络】，而远端文件夹则与本地根文件夹设置为相同的目录。

站点设置中，测试服务器设置如图 8.3 所示。

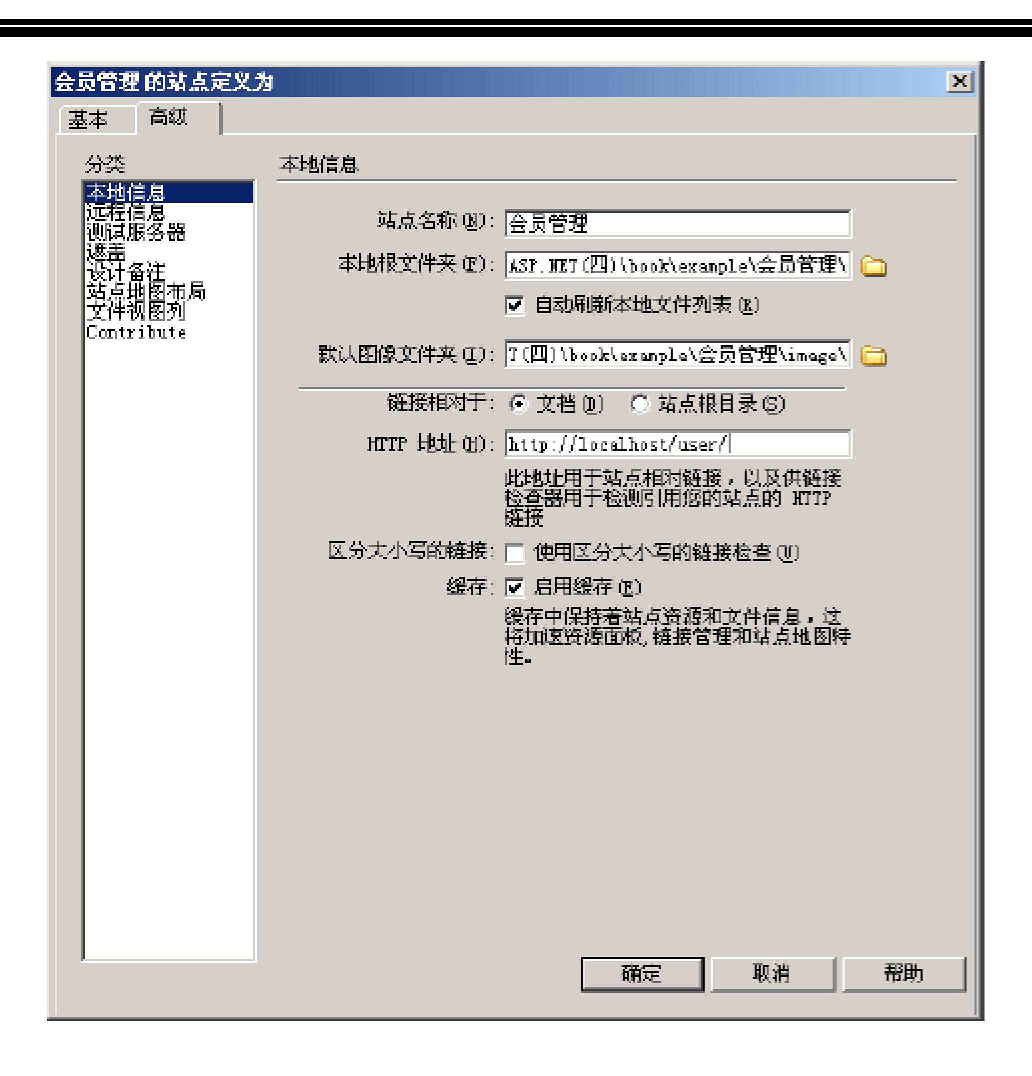


图 8.1　本地信息

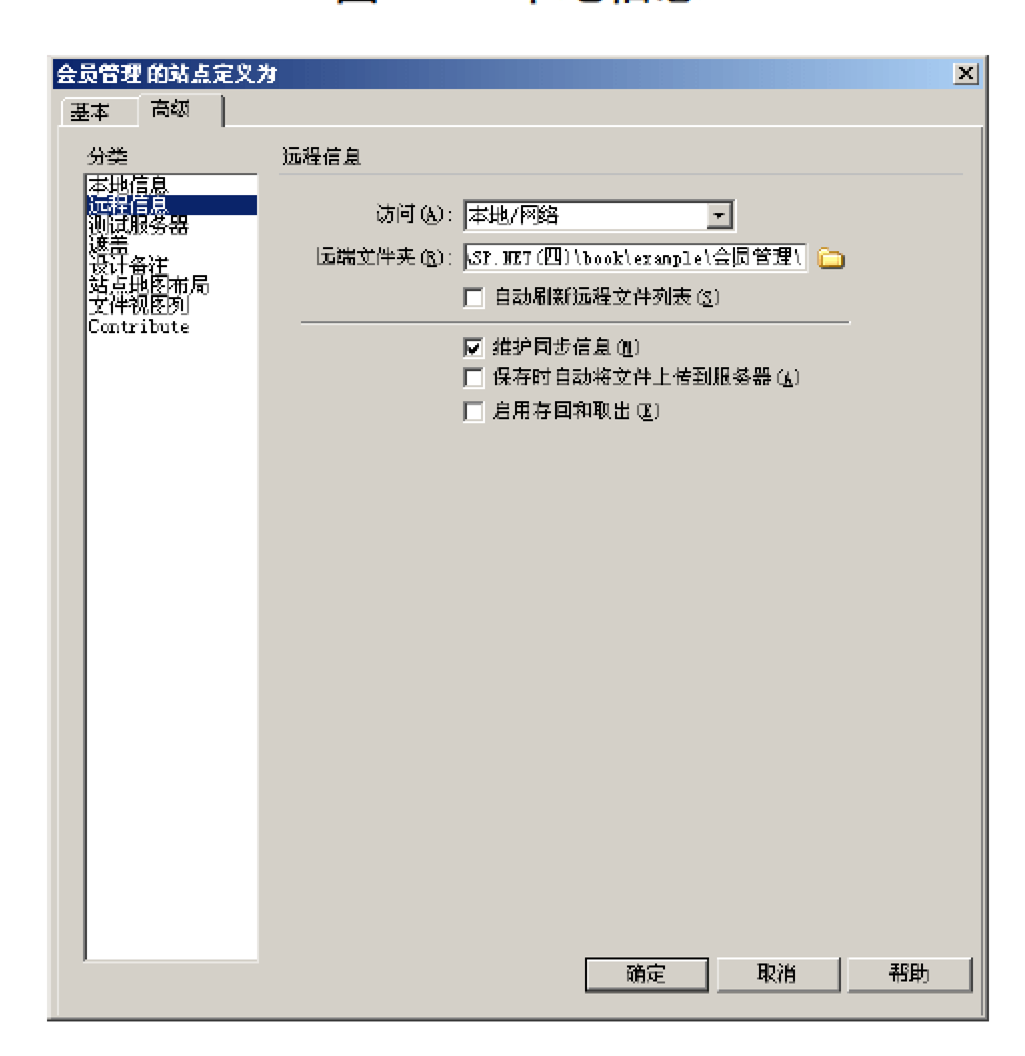


图 8.2　远程信息

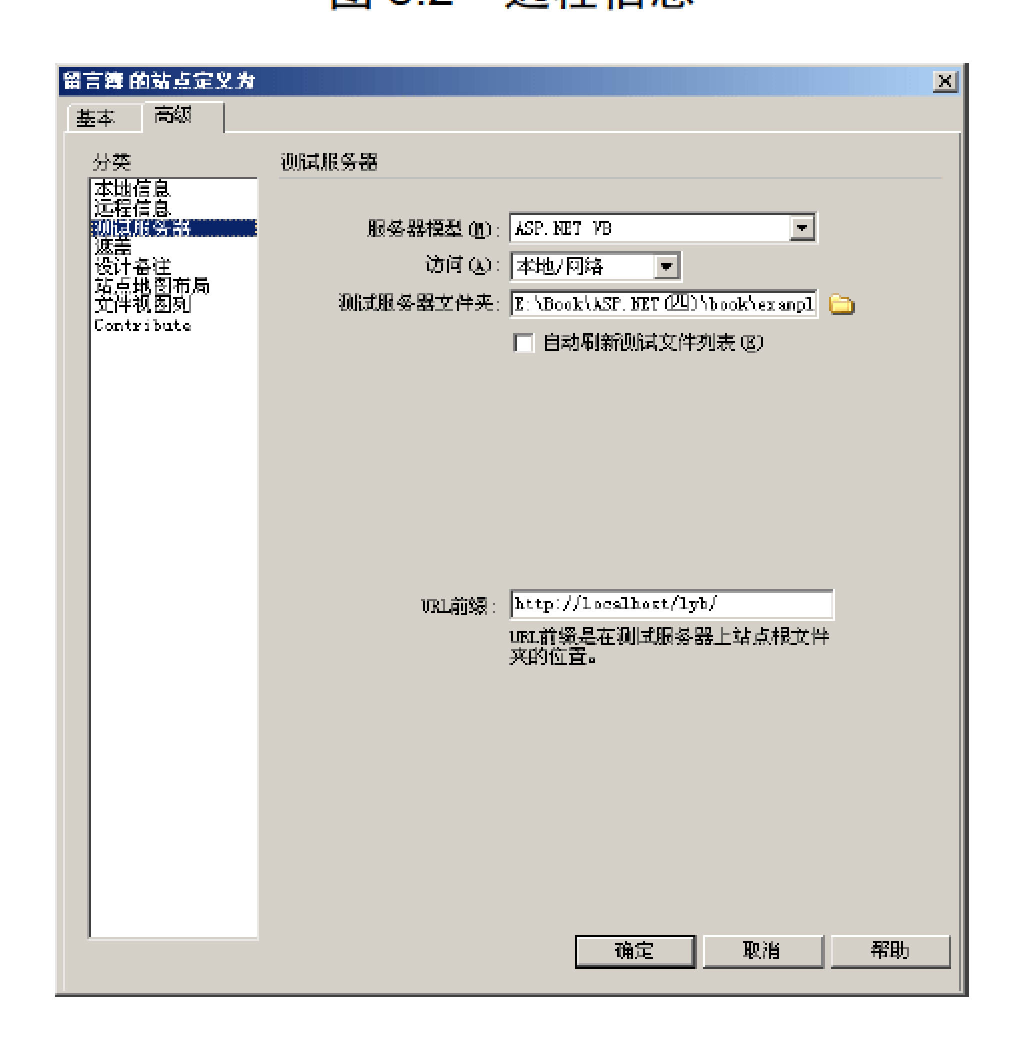


图 8.3　测试服务器

对于站点的部署，在上一章中已经进行了详细的介绍，本章不再赘述。需要记住的是，必须对站点进行部署；否则，系统是无法正常运行的。

8.1.4　定义数据库连接

在介绍各项功能的具体实现之前，要先定义一个数据库连接，这是在 Dreamweaver 8 中进行数据库操作的必要前提。

在 Dreamweaver 8 中打开所定义的站点“会员管理”，新建一个 ASP.NET 页面，打开【应用程序】面板组，选择【数据库】面板，如图 8.4 所示。

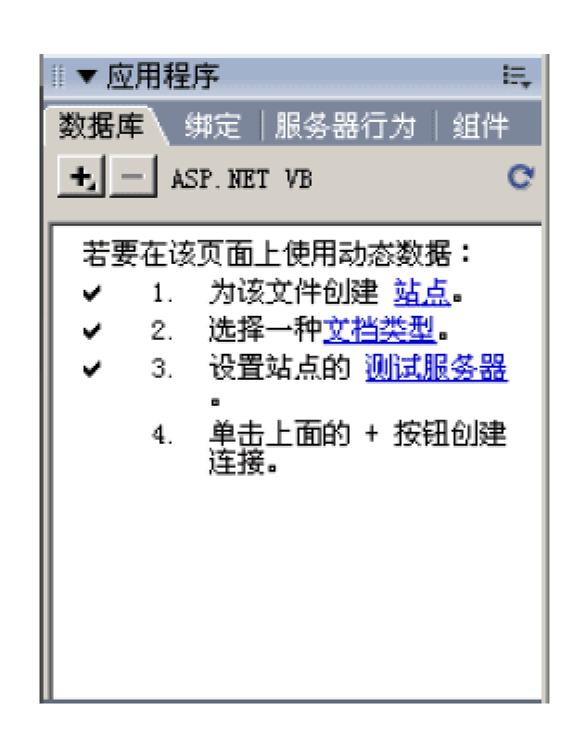


图 8.4　【数据库】面板

单击 **+** 按钮，在弹出的菜单中选择【OLE DB 连接】命令。此时，系统将弹出【OLE DB 连接】对话框，如图 8.5 所示。

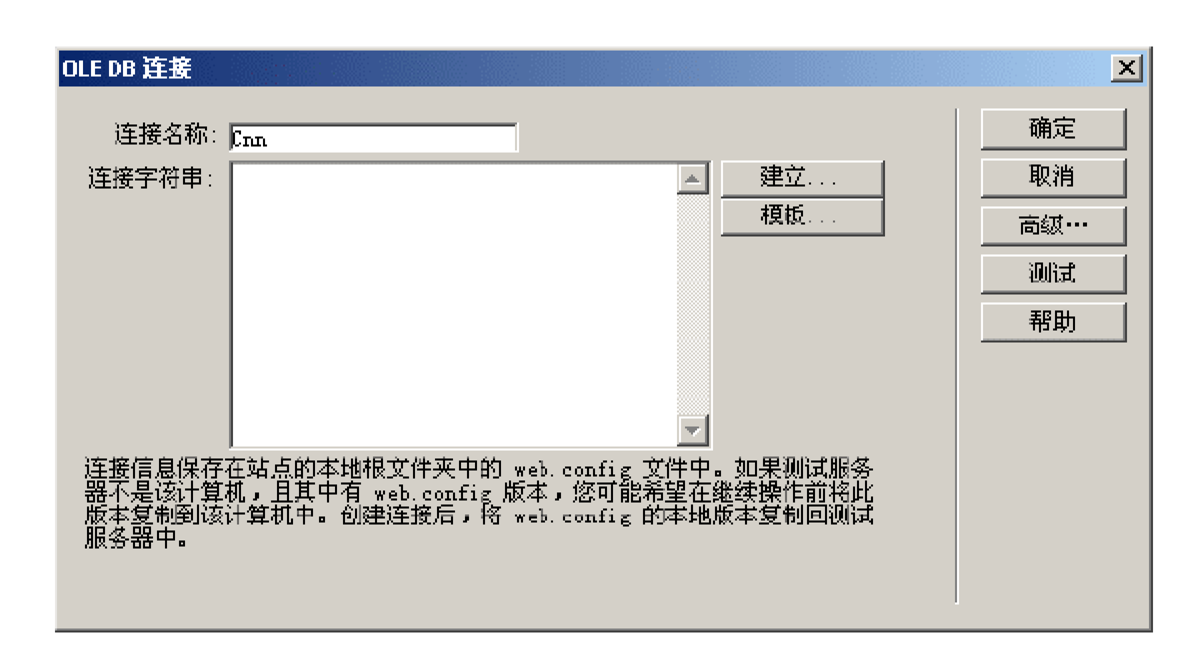


图 8.5　【OLE DB 连接】对话框

在连接名称中输入“Cnn”，单击【建立】按钮，在弹出的 Data Link Properties 对话框中，单击 Provider 标签，选择 Microsoft Jet 4.0 OLE DB Provider。然后，单击 Connection 标签，如图 8.6 所示。

单击【浏览】按钮 ... ，选择会员管理磁盘目录下的 Microsoft Access 数据库 UserInfo.mdb 作为连接的对象。单击【确定】按钮，返回【OLE DB 连接】对话框，并再次单击【确定】按钮，完成数据库连接的创建。此时，在【应用程序】面板组中的【数据库】面板中将会显示用户刚才创建的数据库连接，如图 8.7 所示。

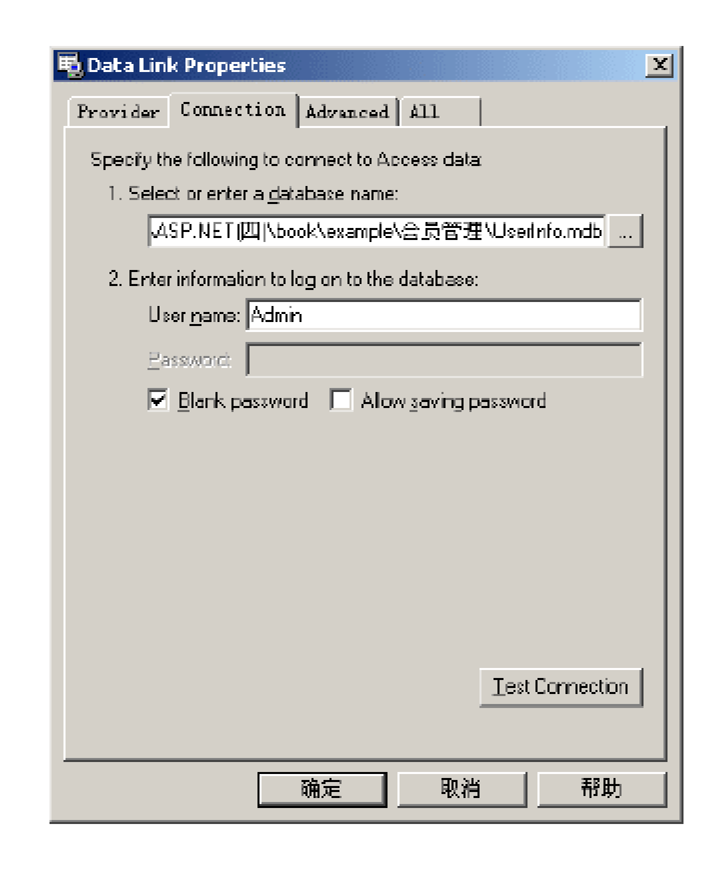


图 8.6　切换至 Connection 选项卡

图 8.7　【数据库】面板

8.2 用 户 注 册

用户注册的整个操作可分为 5 个步骤，首先是站点用户注册的许可协议，在同意协议之后，进入用户资料的填写过程，该页面包括必填信息和选填信息。然后，显示用户所填写的资料，进行资料确认。最后，将数据保存至数据库中，提示注册成功，并发送信息至用户所填写的电子邮箱中，要求用户接收邮件并激活账户。

对于各个步骤操作界面的切换，可通过 ASP.NET 中的 Panel 控件来实现。

8.2.1 许可协议

用户注册时，要获得许可协议，具体操作如下。

(1) 新建一个 ASP.NET 页面，将其保存为 Login.aspx。然后，添加一个表单 Form，设置其 ID 为 Reg，同时切换至【代码】视图，在 Form 中添加属性 “Runat="Server"”。

(2) 切换至【设计】视图，单击【ASP.NET-插入】工具栏中的按钮，在弹出的【标签选择器】对话框中，选择【ASP.NET 标签】|【Web 服务器】分类，并在右边的列表框中选择 asp:Panel，如图 8.8 所示。

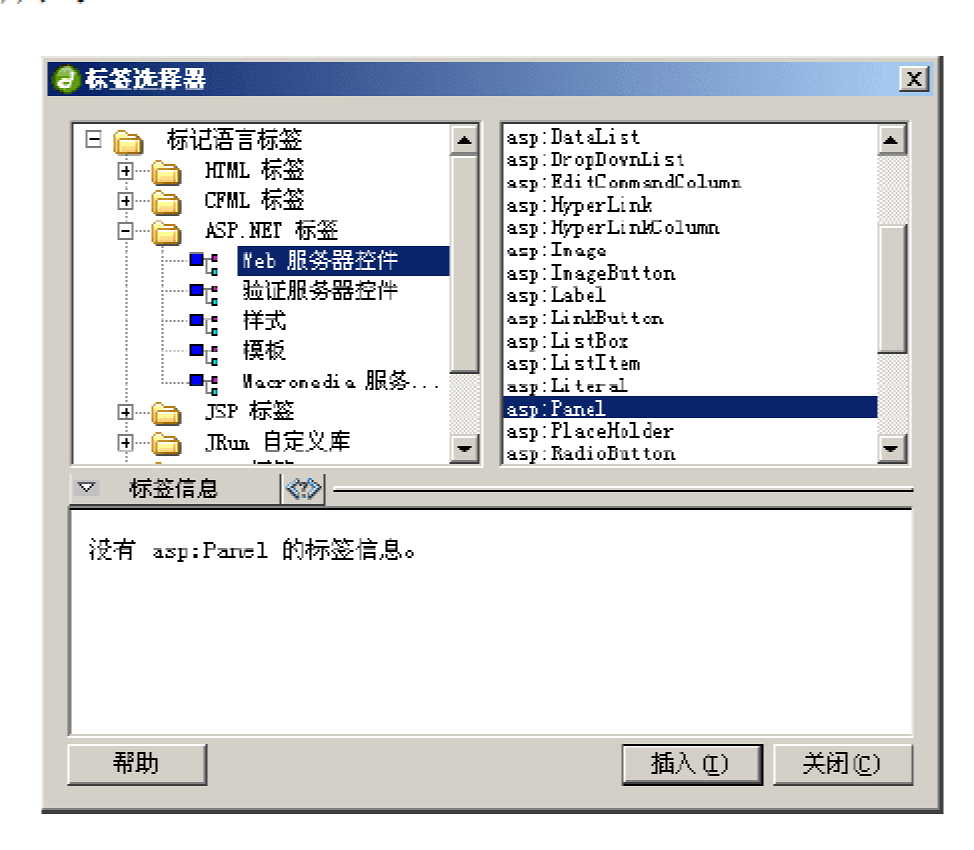


图 8.8 【标签选择器】对话框

将光标置于表单内，单击【插入】按钮，此时将弹出【标签编辑器-Panel】对话框。在【常规】分类选项中，设置其 ID 属性为 Step1，如图 8.9 所示。

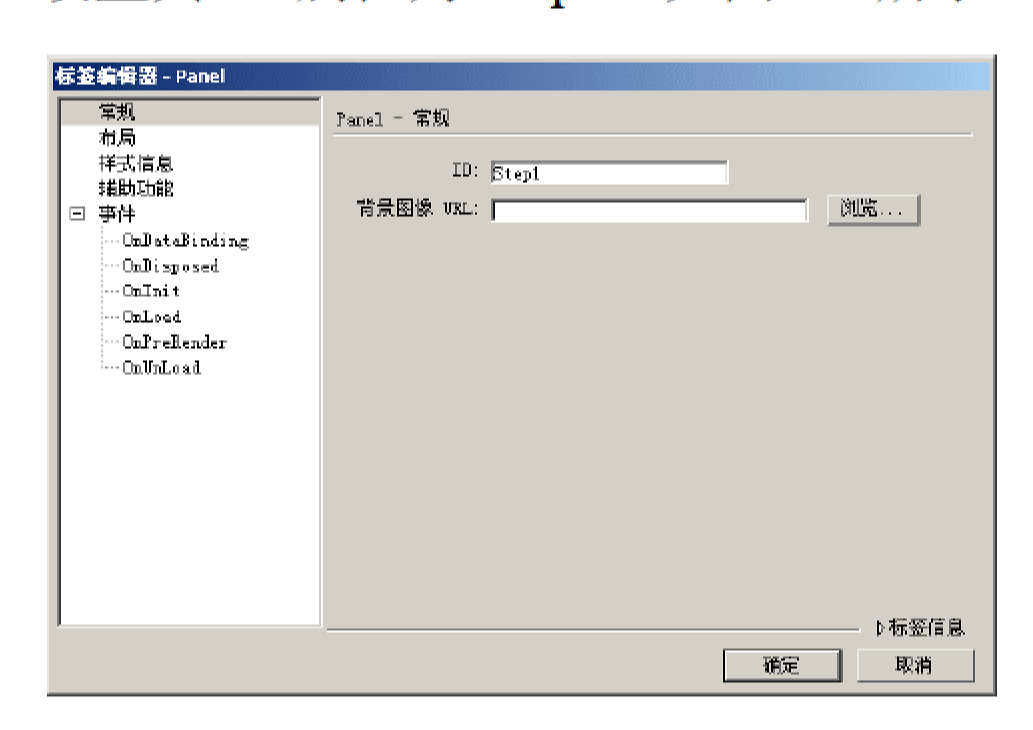


图 8.9 【标签编辑器-Panel】对话框

同时，在【布局】选项中，设置其高度为 300 像素，宽度为 600 像素。单击【确定】按钮，返回【标签选择器】对话框，单击【关闭】按钮，完成 Panel 控件的插入。

(3)　重复以上操作，再次添加 3 个 Panel 控件，分别设置其 ID 为 Step2、Step3 和 Step4。至此，在页面上已经添加了 4 个 Panel 控件，它们分别用于包含用户注册中许可协议、填写个人资料、确认信息和注册成功共 4 个步骤的操作界面。

提示： Panel 服务器控件主要用于为其他控件提供容器。通过设置 Panel 控件的显示或隐藏，可同时显示或隐藏该控件内的所有控件。

(4)　在 Panel 控件 Step1 中，添加一个表格，并添加相关的注册协议，详细操作不再叙述，如图 8.10 所示。

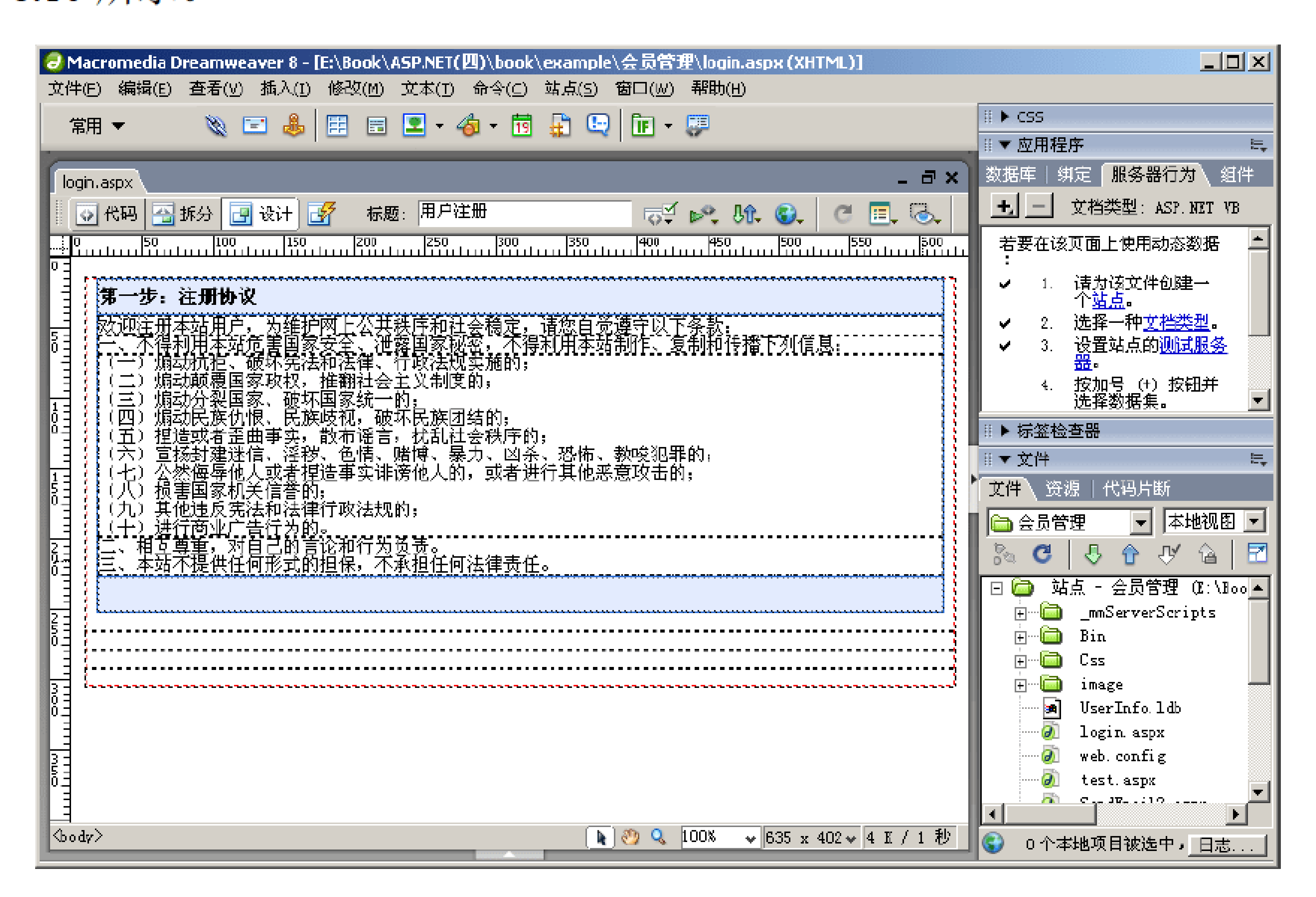


图 8.10　注册协议

在注册协议下，需提供两个操作按钮，分别是【同意】和【不同意】。当单击【同意】按钮时，将进入下一步骤；当单击【不同意】按钮时，将返回用户登录页面。

(5)　在【ASP.NET-插入】工具栏中，拖动按钮至注册协议下的单元格中，在弹出的【asp:按钮】对话框中，设置其 ID 为 Agree，显示文本为“同意”，如图 8.11 所示。

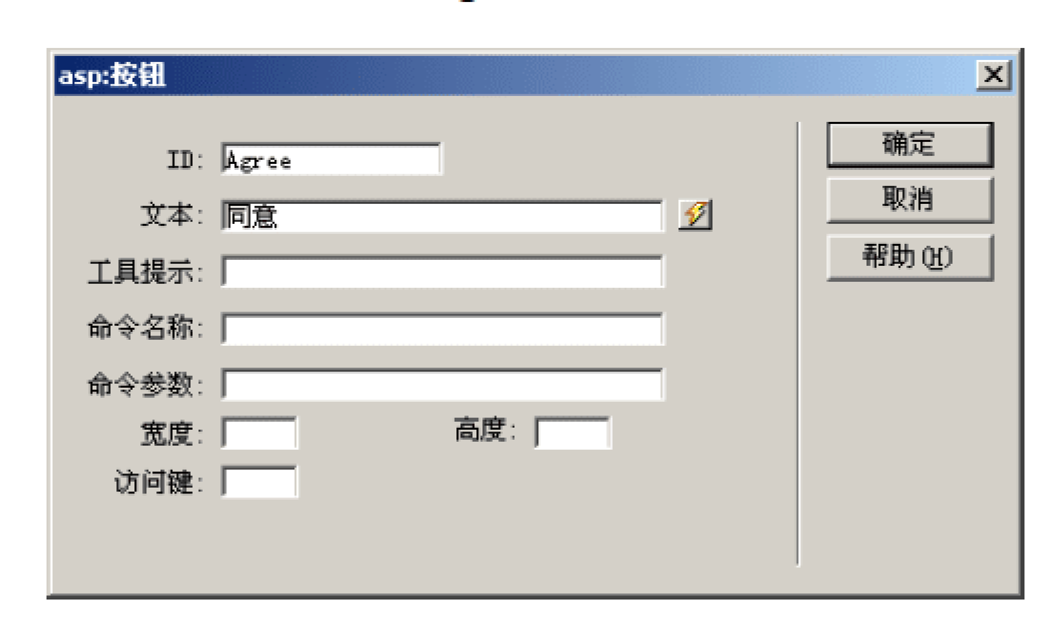


图 8.11　添加按钮控件 Agree

(6)　单击【确定】按钮，完成插入。在页面上选择该按钮并右击，从弹出的菜单中选择【编辑标签】命令。在弹出的【标签编辑器-Button】对话框中，取消【常规】分类下的【原因确认】复选框的选择，如图 8.12 所示。

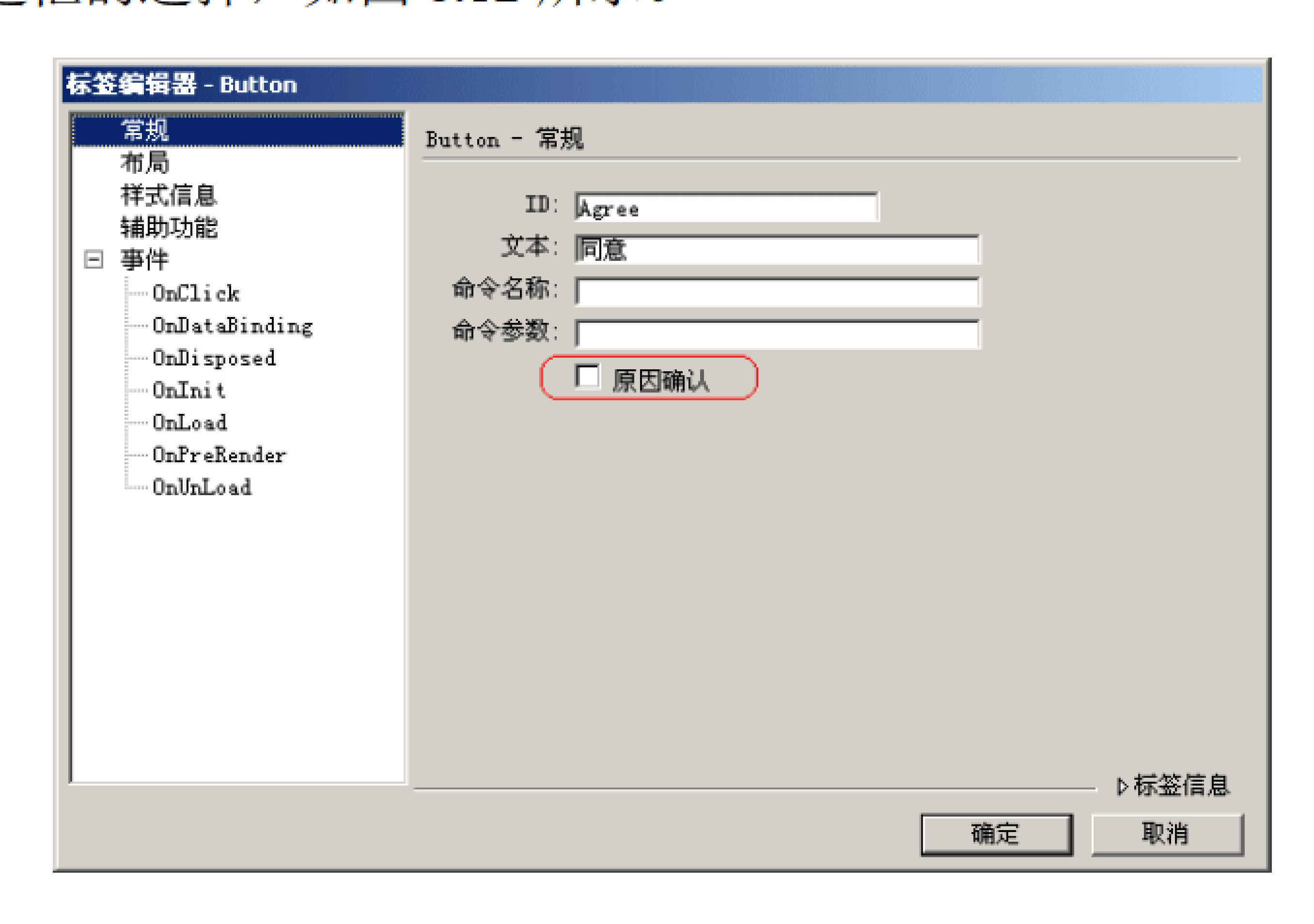


图 8.12　标签编辑器

【原因确认】复选框对应的是按钮控件的 CausesValidation 属性。该属性用于指示当单击该按钮时是否执行页面验证。由于在用户注册的第二步(即填写个人资料)中，将会设置相应的验证控件来对用户的输入进行验证。因此，这里将该按钮的 CausesValidation 属性设置为 Flase，以避免由于页面验证不通过而导致无法正常执行按钮的单击事件代码。

(7)　选择【事件】|OnClick 分类，在右侧的文本框中输入事件名 Agree_Click，如图 8.13 所示。

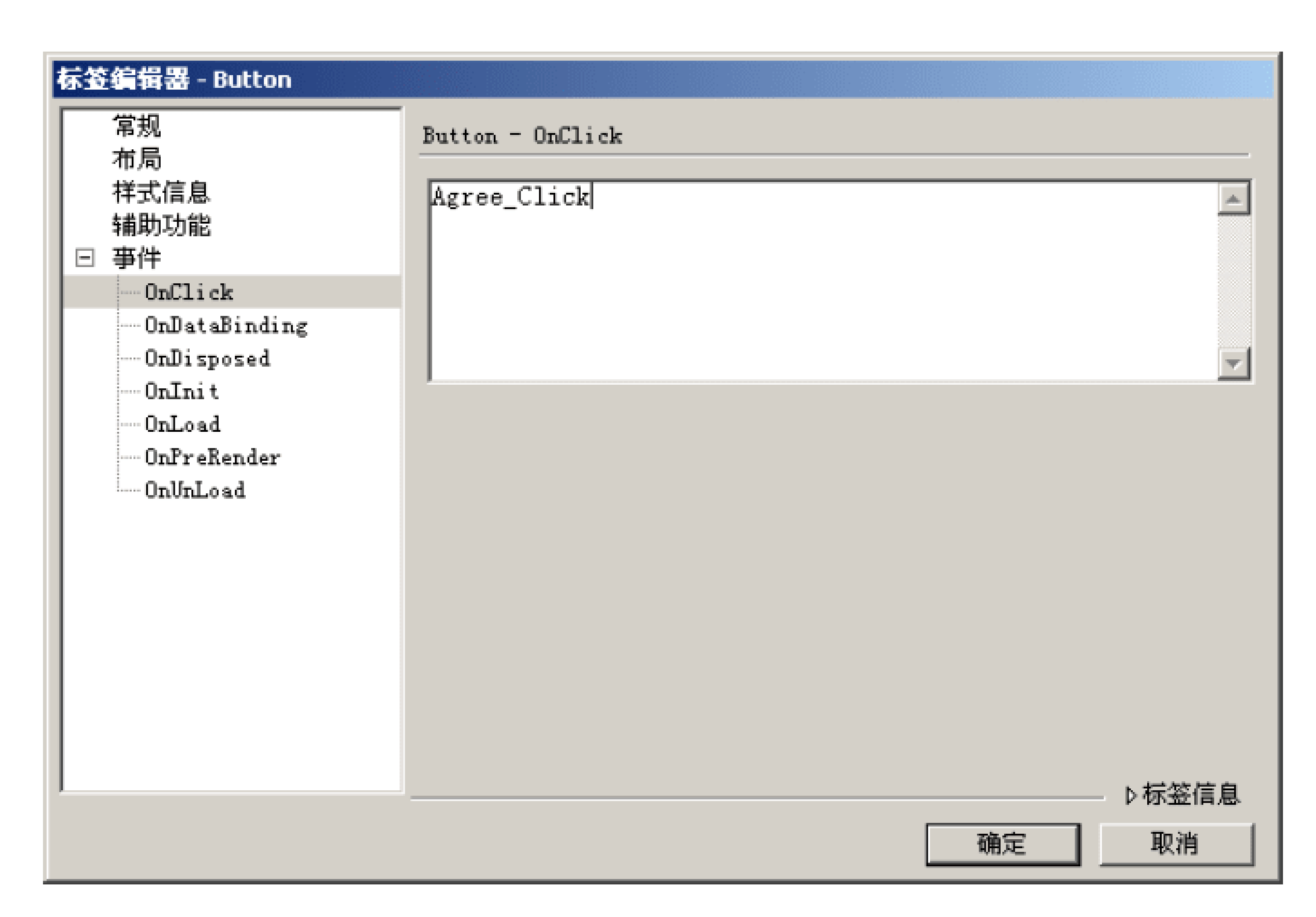


图 8.13　设置按钮的单击事件

这里，设置了按钮 Agree 的单击事件为 Agree_Click，该事件代码将在【代码】视图中进行编写。

(8)　重复以上步骤，添加一个新的按钮控件，设置其 ID 为 NotAgree，显示文本为“不同意”。同时，在【标签编辑器】对话框中，取消该按钮的【原因确认】复选框的选择，

设置其单击事件为 NotAgree_Click。

(9) 切换至【代码】视图，编写页面加载时所执行的 Page_Load 事件及各按钮的单击事件，代码如下：

【示例代码】

```
<script runat="server">
  '添加 Page_Load 事件
  Sub Page_Load(Sender As Object,E As EventArgs)
    Dim i As Integer
    if (not Page.IsPostBack)
      '设置 Step1 可见，其他的 Panel 控件隐藏
      step1.visible=True
      step2.visible=false
      step3.visible=false
      step4.visible=false
    end if
  End Sub
 '添加【同意】按钮的单击事件
  Sub Agree_Click(Sender As Object,E As EventArgs)
    '设置 Step2 可见，其他的 Panel 控件隐藏
    step1.visible=false
    step2.visible=true
    step3.visible=false
    step4.visible=false
  End Sub
  '添加【不同意】按钮的单击事件
  Sub NotAgree_Click(Sender As Object,E As EventArgs)
    '将页面跳转至用户登录页面
    page.response.redirect("default.aspx")
  End Sub
</script>
```

此时，用户注册的第 1 步操作设计完成，预览效果如图 8.14 所示。

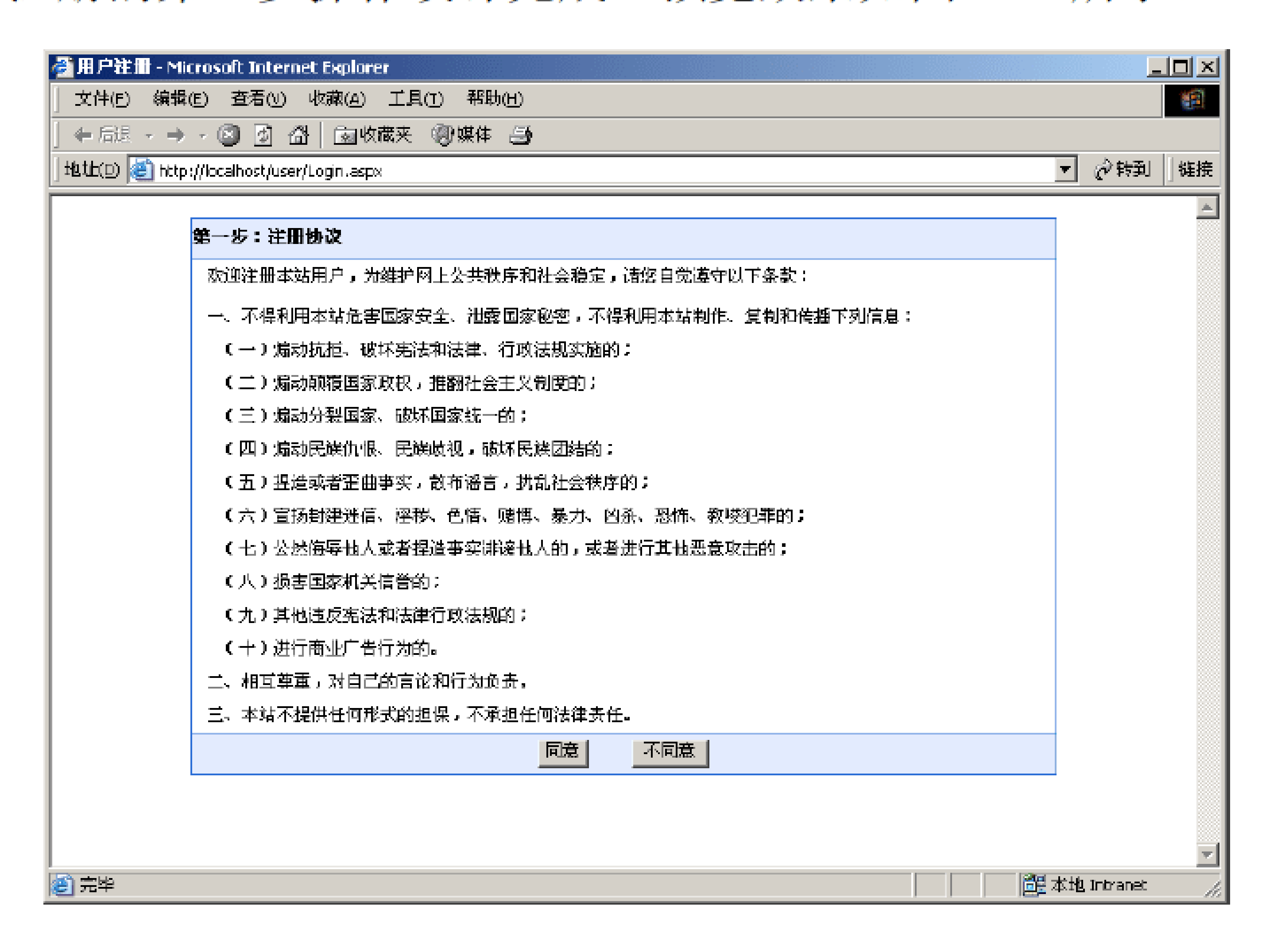


图 8.14　页面预览效果

当单击【同意】按钮时，将进入下一步操作；当单击【不同意】按钮时，将返回用户登录页面。

8.2.2　填写个人资料

下面，来看用户注册中第 2 步操作的设计，即填写个人资料。在此步操作中，主要是提供各种文本框或选择框以供用户输入或选择。

1. 设计填写人个资料的功能界面

在 ID 为 Step2 的 Panel 控件中，添加一个表格，并插入各种文本和相关输入控件，如图 8.15 所示。

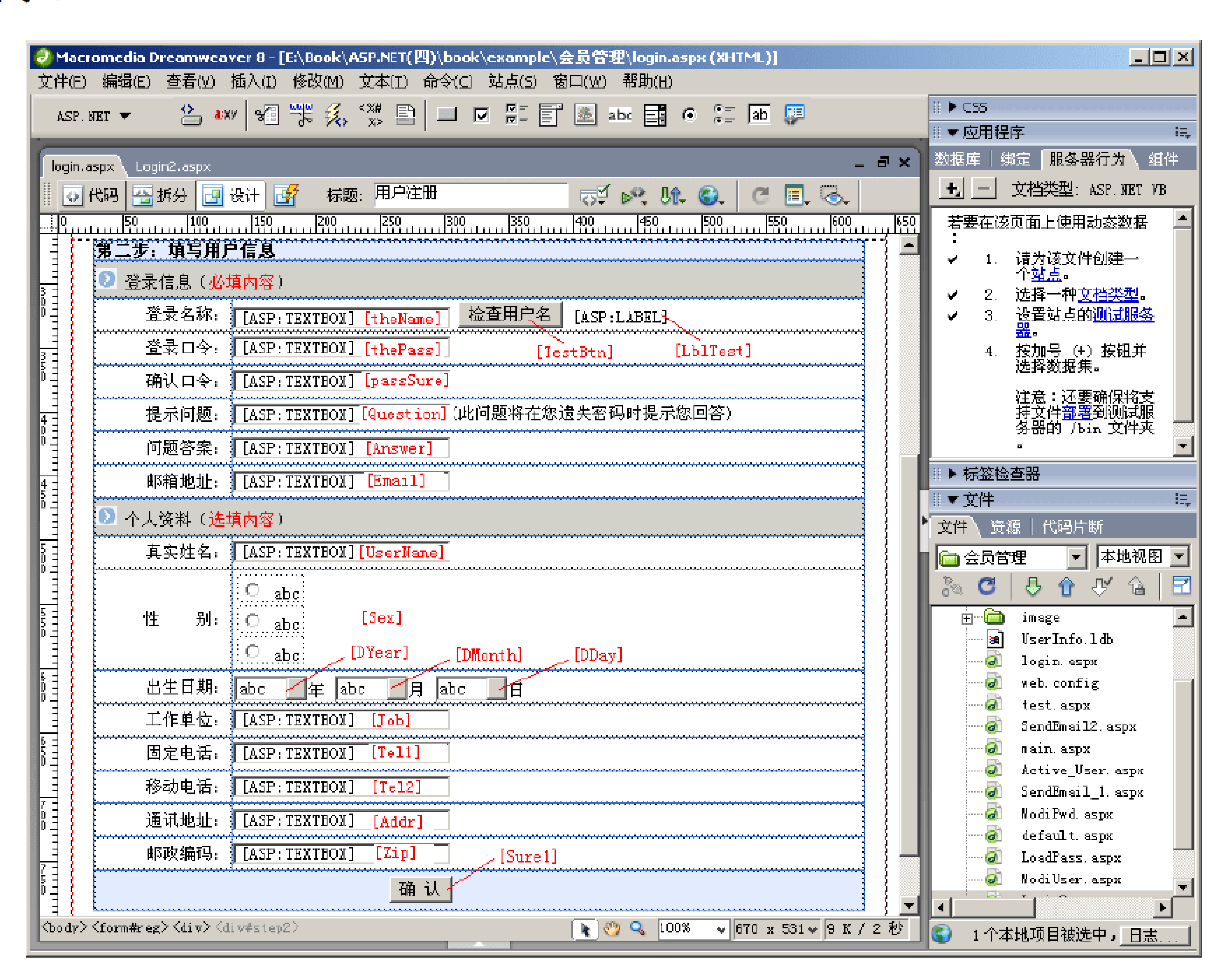


图 8.15　填写个人资料功能界面

在图中，用文字标记了各个文本框控件和下拉列表框控件所对应的 ID 名称。这里，不再对各个控件的插入操作进行详细的叙述。对于其中部分控件说明如下。

- 对于登录口令和确认口令对应的文本框 thePass 和 PassSure，其文本框类型均应设置为【密码】。
- 性别选项对应的是单选按钮列表控件(RadioButtonList 控件)，其中的选项包括“男”和“女”，其代码定义如图 8.16 所示。

```
<TR >
  <TD align="right" bgcolor="white">性    别: </TD>
  <TD vAlign="top" bgcolor="white" style="padding-left:5px;">
    <asp:RadioButtonList id="Sex" Runat="server" RepeatDirection="Horizontal">
      <asp:ListItem Value="男" Selected="True">男</asp:ListItem>
      <asp:ListItem Value="女">女</asp:ListItem>
    </asp:RadioButtonList>
  </TD>
</TR>
```

图 8.16　定义 RadioButtonList 控件

- 对于出生日期中的年、月、日三个下拉列表框控件的数据绑定，是在 Page_Load 事件中进行加载的。其中，年度选择为 1920—2006，月份选择为 1～12，日的选择为 1～31，如图 8.17 所示。

```
Sub Page_Load(Sender As Object,E As EventArgs)
  Dim theYear As new ArrayList
  Dim theMonth As new ArrayList
  Dim theDay As new ArrayList
  Dim i As Integer
  if (not Page.IsPostBack)
    step1.visible=True
    step2.visible=false
    step3.visible=false
    step4.visible=false
    For i = 1920 To 2006
      theYear.Add(i.ToString())
    Next
    For i=1 To 12
      theMonth.Add(i.toString())
    Next
    For i=1 To 31
      theDay.Add(i.toString())
    Next
    DYear.DataSource = theYear
    dyear.databind()
    dmonth.datasource=themonth
    dmonth.databind()
    dday.datasource=theday
    dday.databind()
  end if
End Sub
```

定义三个数组，分别用于存储年、月、日的选择数据
填充年的选择数据
填充月的选择数据
填充日的选择数据
分别绑定年、月、日三个下拉框控件的数据显示

图 8.17　绑定年、月、日的选择数据

- 对于不同的月份，其可供选择的日是不同的。例如，对于 1 月、3 月、5 月、7 月、8 月、10 月、12 月，其日的选择是 1～31；对于 4 月、6 月、9 月、11 月，其日的选择是 1～30；而对于 2 月，则还必须判断用户所选择的年是否为闰年，如果为闰年，则日的选择为 1～29，否则为 1～28。也就是说，日的选择范围是根据用户所选择的年和月来动态决定的。因此，将月所对应的 DropDownList 控件 Dmonth 的 AutoPostBack(自动回发)属性设置为 True，同时设置其 OnSelectedIndexChanged 事件为 ChangeDataMon，该事件需在代码中进行编写，代码如下：

【示例代码】

```
Sub ChangeDateMon(ByVal Sender As Object, ByVal E As EventArgs)
  '定义各月份所对应的天数，其中 2 月份暂定为 28 天，即非闰年的情况
  Dim Days As Integer() = {31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31}
  Dim i As Integer
  '定义数据 theDay，用于存储当前用户所选择的月份对应的天
  Dim theDay As New ArrayList
  '判断当前用户所选择的年度是否为闰年
  If ((Convert.ToInt32(DYear.SelectedItem.Text.ToString()) Mod 4) = 0
And (Convert.ToInt32(DYear.SelectedItem.Text.ToString()) Mod 100) <> 0)
Then
    '当前年度为闰年，修改数组，将 2 月份所对应的天数改为 29 天
    Days(1) = 29
  End If
  '获取当前用户所选择的月份所对应的天数，并通过循环向数组 theDay 中添加数据
  For i = 1 To Days(Convert.ToInt32(DMonth.SelectedItem.Text.ToString() - 1))
    TheDay.Add(i.ToString)
  Next
```

```
    '将数组 theDay 绑定至日所对应的下拉列表框控件 Dday 中
    DDay.DataSource = theDay
    DDay.DataBind()
  End Sub
```

- 在填写个人资料的操作界面上，提供了两个操作按钮【检查用户名】和【确认】，它们所对应的单击事件分别为 TestUser_Click 和 Sure1_Click，其事件代码将在后面进行叙述。

2. 页面验证控件的添加

这也是本节中的一个重点。所谓验证控件，主要用来对用户的输入进行检测，判断其是否符合所定义的规则。如果用户输入了不符合规则的数据，则不允许用户提交此页面。

在此页面中，需要进行数据验证的控件包括必填内容中的所有文本框控件和邮政编码所对应的文本框控件，其控件 ID 名称分别为 theName、thePass、passSure、Question、Answer、Email 和 Zip。

下面看一下登录名称对应的文本框控件 theName。该控件需要绑定两种验证控件：一种是必须输入验证控件(RequiredFieldValidator)，另一种是正则表达式验证控件(RegularExpressionValidator)。

(1) 切换至【设计】视图，将光标置于标签控件 LblTest 之后，单击【ASP.NET-插入】工具栏中的按钮，在弹出的【标签选择器】对话框中，选择【ASP.NET 标签】|【验证服务器控件】分类，并在右边的列表框中选择 asp:RequiredFieldValidator，如图 8.18 所示。

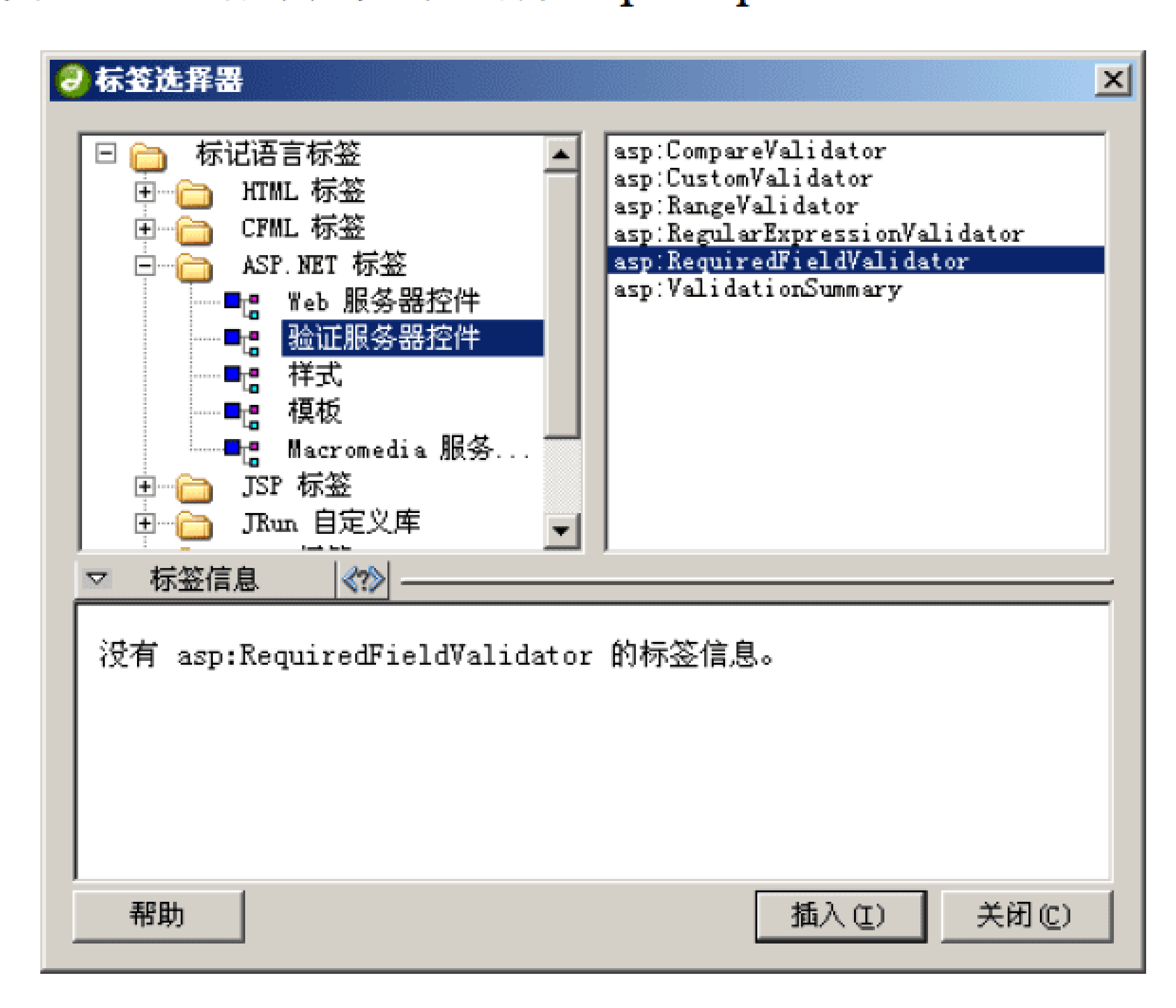


图 8.18　【标签选择器】对话框

(2) 单击【插入】按钮，在弹出的【标签编辑器】对话框中，设置其 ID 为 Required1，显示文本为“请输入登录名称！”，显示模式为“动态”，要验证的控件为 theName，错误信息为“请输入登录名称！”，如图 8.19 所示。

(3) 单击【确定】按钮，返回【标签选择器】对话框，再单击【关闭】按钮，完成该验证控件的插入。此控件为必须输入控件，要求用户必须在文本框 theName 中输入数据。

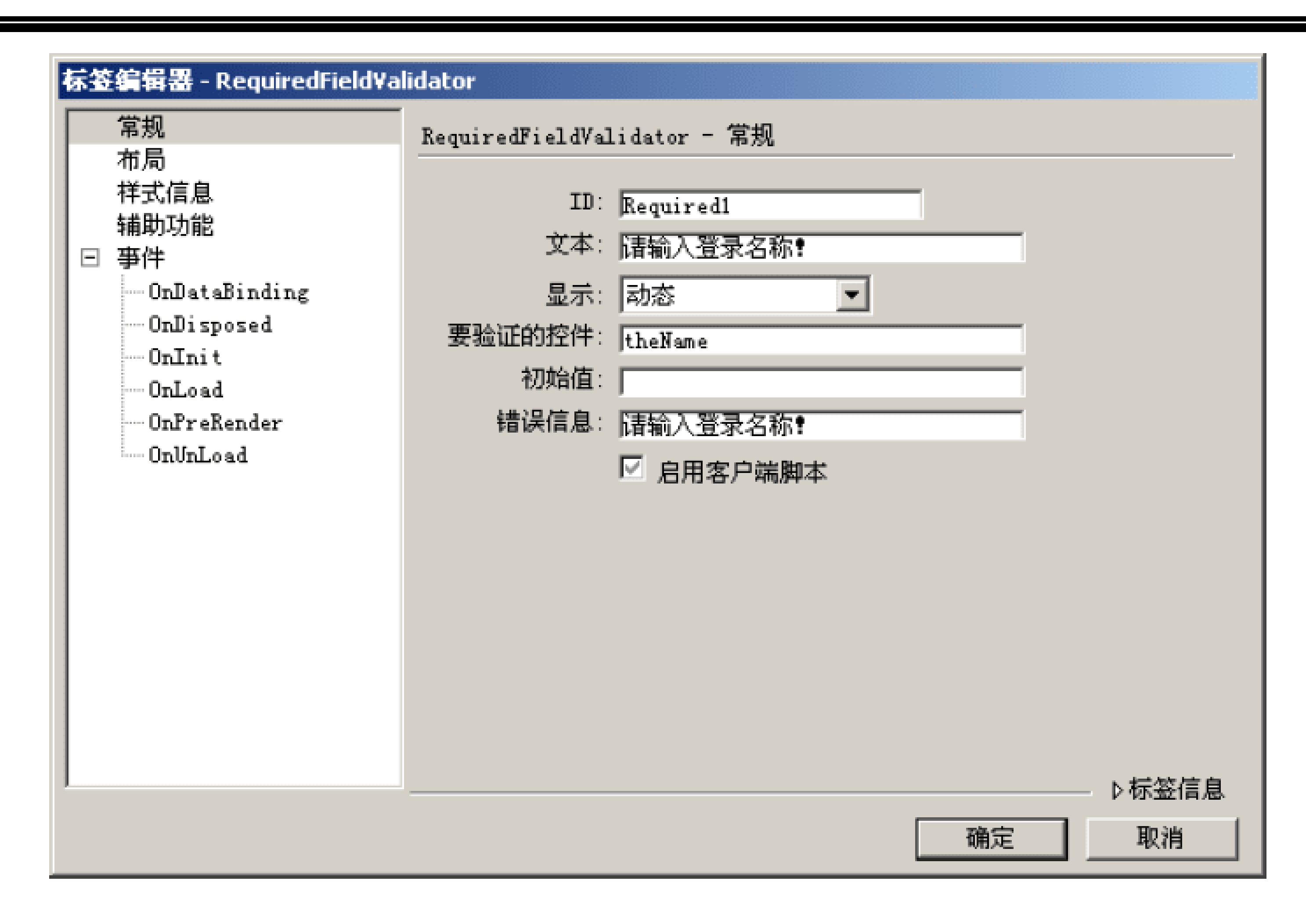


图 8.19　设置验证控件 Required1 的属性

(4) 将光标置于验证控件 Required1 之后，再次单击【ASP.NET-插入】工具栏中的按钮，在弹出的【标签选择器】对话框中，选择【ASP.NET 标签】|【验证服务器控件】分类，并在右边的列表框中选择 asp:RegularExpressionValidator，单击【插入】按钮，此时将弹出【标签编辑器】对话框，如图 8.20 所示。

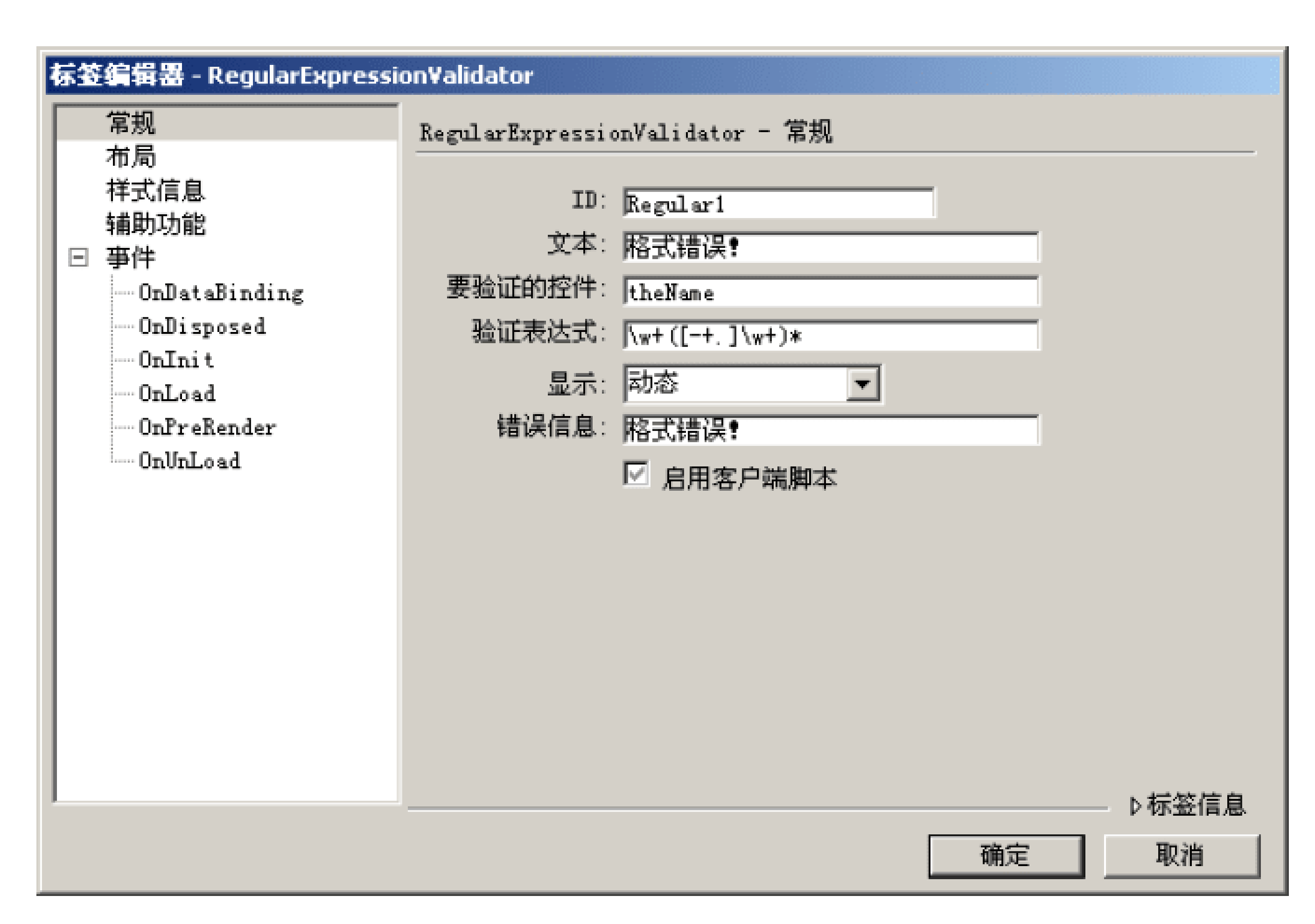


图 8.20　设置验证控件 Regular1 的属性

(5) 在该对话框中，设置其 ID 为 Regular1，文本为“格式错误！”，要验证的控件为“theName”，验证表达式为“\w+([-+.]\w+)*”，显示模式为“动态”，错误信息为“格式错误！”。这里设置了一个正则表达式，来禁止用户在文本框 theName 中输入非法字符。

(6) 对于登录口令所对应的文本框控件 thePass，需添加关联的必须输入验证控件(RequiredFieldValidator)，其属性设置如图 8.21 所示。

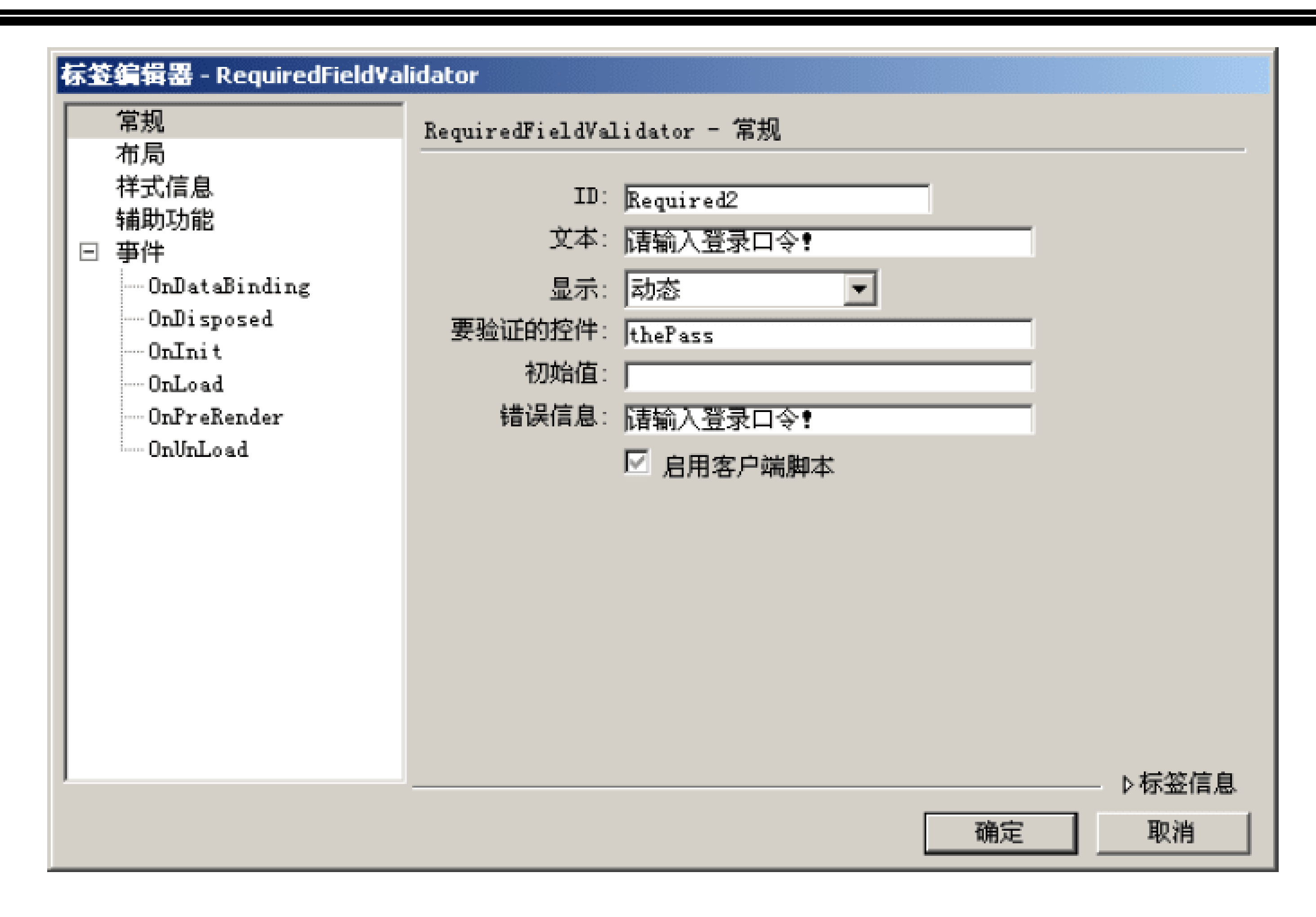


图 8.21　设置验证控件 Required2 的属性

对于确认口令所对应的文本框控件 PassSure，需添加关联的比较验证控件(CompareValidator)，其属性设置如图 8.22 所示。

图 8.22　设置验证控件 Compare1 的属性

这里所设置的验证的控件，用于比较确认口令与用户口令所对应的文本框的值是否一致。如果不一致，则验证失败，提示错误信息。

对于提示问题所对应的文本框控件 Question，需添加关联的必须输入验证控件(RequiredFieldValidator)，其属性设置如图 8.23 所示。

对于问题答案所对应的文本框控件 Answer，需添加关联的必须输入验证控件(RequiredFieldValidator)，其属性设置如图 8.24 所示。

对于邮箱地址所对应的文本框控件 Email，除了需要添加关联的必须输入验证控件(RequiredFieldValidator)外，还需添加正则表达式验证控件(RegularExpressionValidator)以保证其输入的正确性，其属性设置分别如图 8.25 和图 8.26 所示。

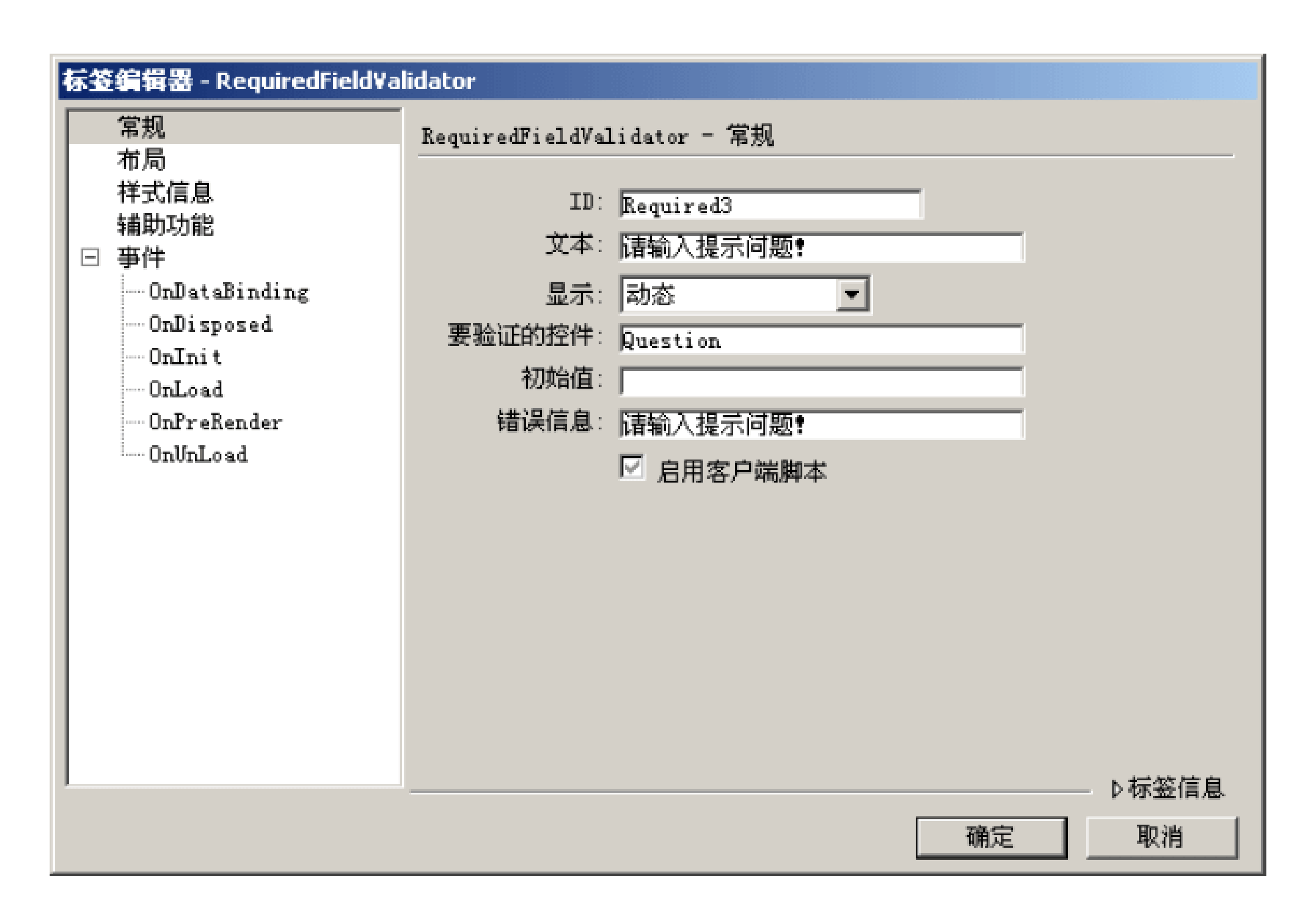


图 8.23　设置验证控件 Required3 的属性

图 8.24　设置验证控件 Required4 的属性

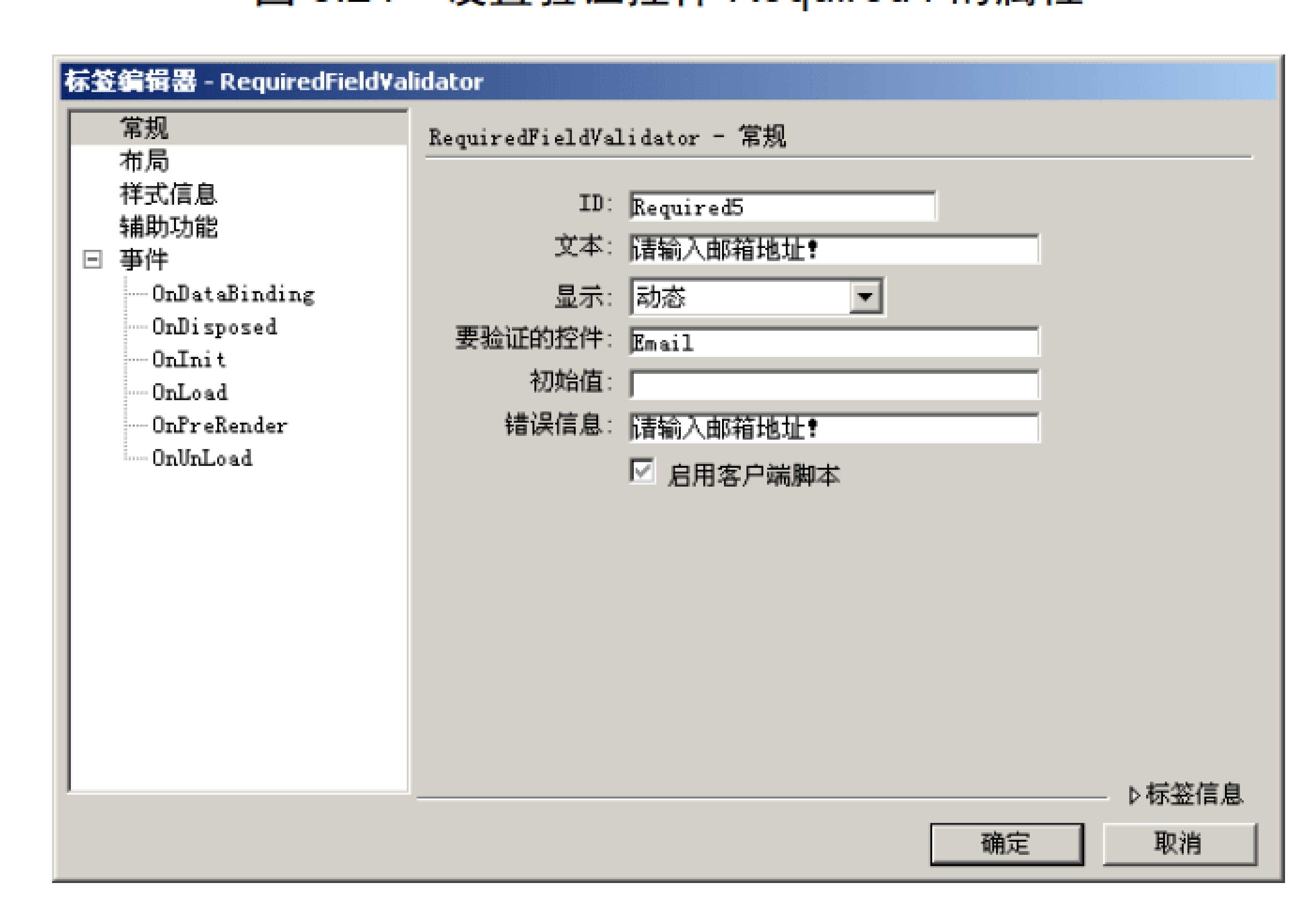


图 8.25　设置验证控件 Required5 的属性

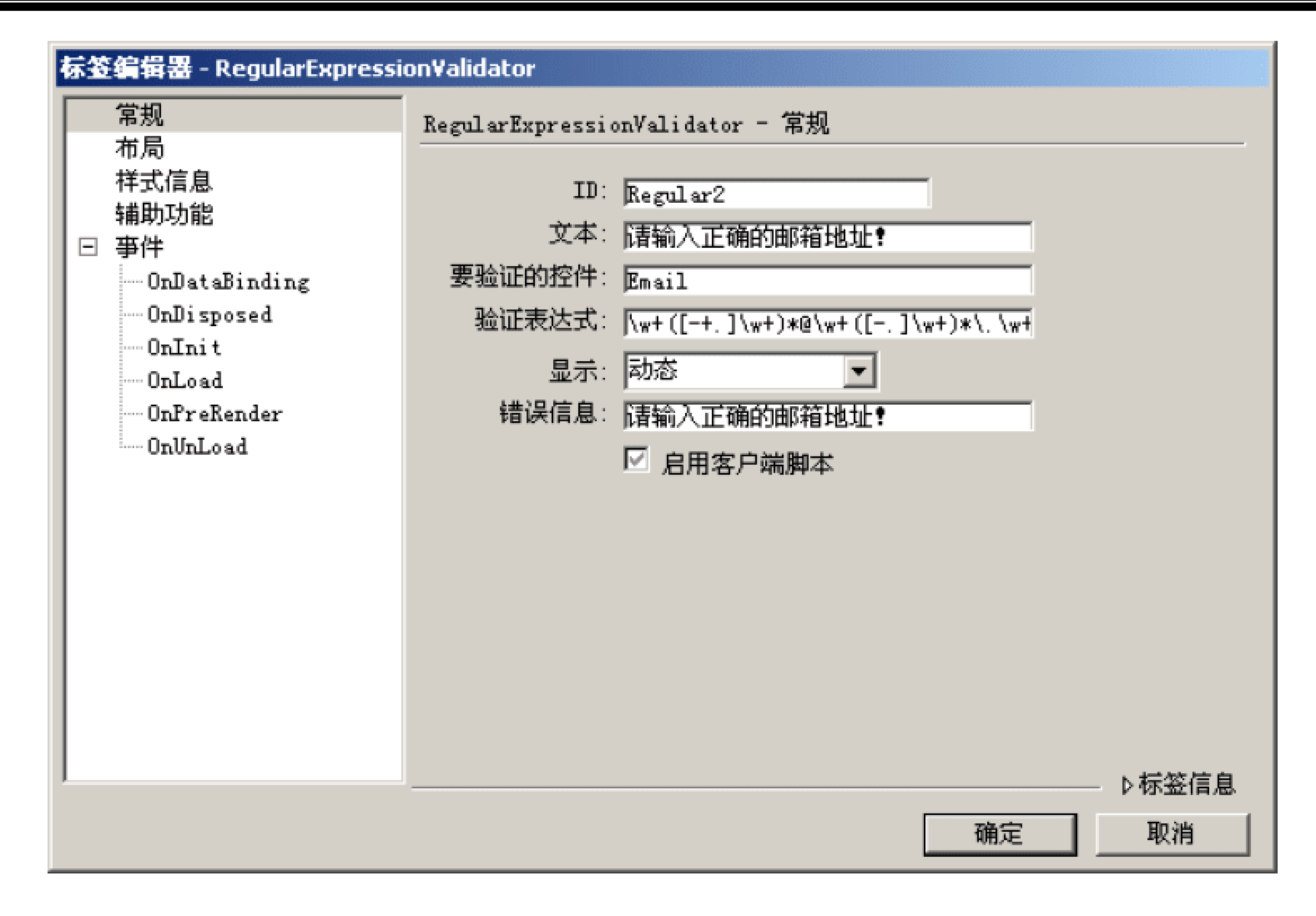


图 8.26　设置验证控件 Regular2 的属性

对于邮政编码所对应的文本框控件 Zip，需添加正则表达式验证控件(RegularExpressionValidator)以保证其输入的正确性，其属性设置分别如图 8.27 所示。

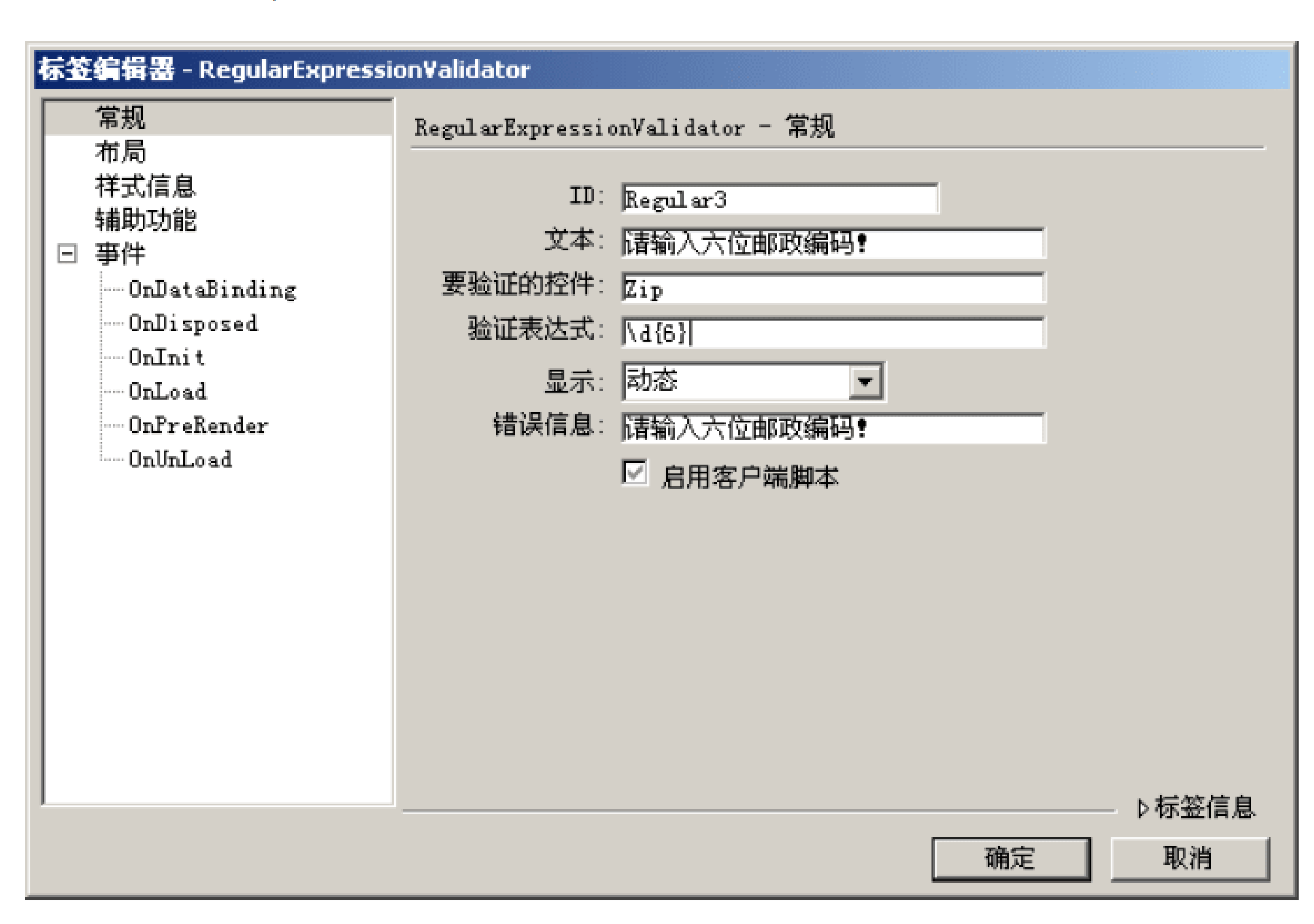


图 8.27　设置验证控件 Regular3 的属性

至此，填写个人资料功能的结构设计完成。

3. 【检测用户名】按钮的单击事件

【检测用户名】按钮主要用于检测用户所输入的登录名称是否已被他人注册，这里可以通过创建数据集来实现。

(1) 打开【应用程序】面板组，切换至【服务器行为】面板，单击+按钮，在弹出的菜单中选择【数据集】命令，将弹出【数据集】对话框，如图 8.28 所示。

(2) 设置数据集的名称为 DataSet1，选择连接 Cnn，选择表格 UserSheet，在筛选条件中设置字段 LoginName 等于表单变量 theName。单击【确定】按钮，完成数据集的创建。

图 8.28　创建数据集

(3) 将视图切换至【代码】视图，添加【检测用户名】按钮的单击事件 TestUser_Click，代码如下：

【示例代码】

```
Sub TestUser_Click(Sender As Object,E As EventArgs)
  '判断当前用户是否输入了登录名称
  if trim(theName.Text)<>"" then
    '输入了登录名称，判断数据集是否有记录
    if dataset1.defaultView.Table.Rows.Count>0 then
      '存在记录，登录名称已被占用
      Lbltest.text="该用户名已被他人占用！"
    else
      '不存在记录，登录名称尚未占用
      Lbltest.text="该用户名尚未被占用！"
    end if
  else
    '没有输入登录名称，不进行判断，并将标签控件 LblTest 的值清空
    Lbltest.text=""
  end if
end Sub
```

在以上代码中，首先对用户是否输入了登录名称进行判断。如果输入了登录名称，则获取数据集 DataSet1 筛选的结果。如果记录数不为零，说明该名称已在数据库中存在，则提示用户名已占用；如果数据集中没有记录，说明没有查询出结果，提示用户名尚未被占用。如果用户没有输入登录名称，此时无须对数据集进行判断，直接将提示文本标签 LblTest 清空即可。

4. 【确定】按钮的单击事件

事实上，【确定】按钮在此并不执行保存数据的操作，它仅仅是将用户所填写的个人资料信息赋值给“确认信息”功能中的各个标签文本，并完成填写个人资料的操作，进入用户注册的下一步骤：确认信息。而数据的保存操作，也将在“确认信息”步骤中进行。以下是【确定】按钮的单击事件代码：

【示例代码】

```
Sub Sure1_Click(Sender As Object,E As EventArgs)
  LblLoginName.Text = theName.Text  '显示登录名称
  LblPwd.Text =thePass.Text  '显示登录密码
  LblQuestion.Text =Question.Text  '显示提示问题
  LblAnswer.Text = Answer.Text  '显示问题答案
  LblName.Text = UserName.Text  '显示真实名称
  LblSex.Text = Sex.SelectedItem.Text  '显示性别
  LblBirth.Text = DYear.SelectedItem.Text & "-" & DMonth.SelectedItem.Text
& "-" & DDay.SelectedItem.Text  '显示出生日期
  LblWork.Text = job.Text  '显示工作单位
  LblTel.Text = tel1.Text  '显示固定电话
  LblTel1.Text = tel2.Text  '显示移动电话
  LblAddr.Text = addr.Text  '显示联系地址
  LblZip.Text = zip.Text  '显示邮政编码
  Lblemail.Text = email.Text  '显示邮箱地址
  step1.Visible = False  '隐藏第 1 步操作界面
  step2.Visible = False  '隐藏第 2 步操作界面
  step3.Visible = True   '显示第 3 步操作界面
  step4.Visible = False  '隐藏第 4 步操作界面
  LblErr.Text=""  '清空“确认信息”步骤中的错误信息
End Sub
```

至此，用户注册的第 2 步操作设计完成，预览效果如图 8.29 所示。

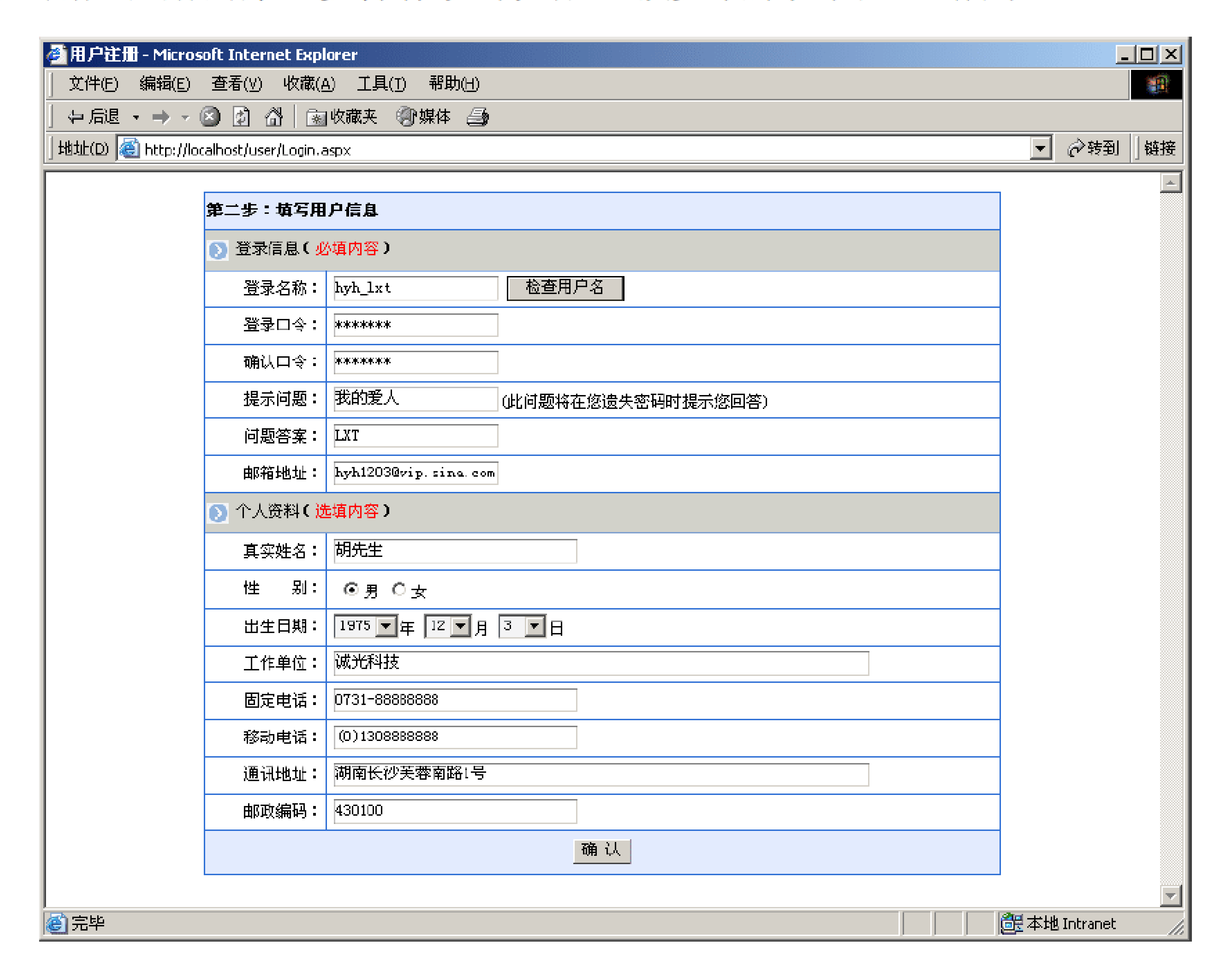


图 8.29　页面预览效果

单击【确认】按钮，可进入下一步操作：确认信息。但是，当用户信息输入不完全，或输入不规范而导致页面无法通过验证时，系统将会自动显示错误信息，而无法跳过当前操作，如图 8.30 所示。

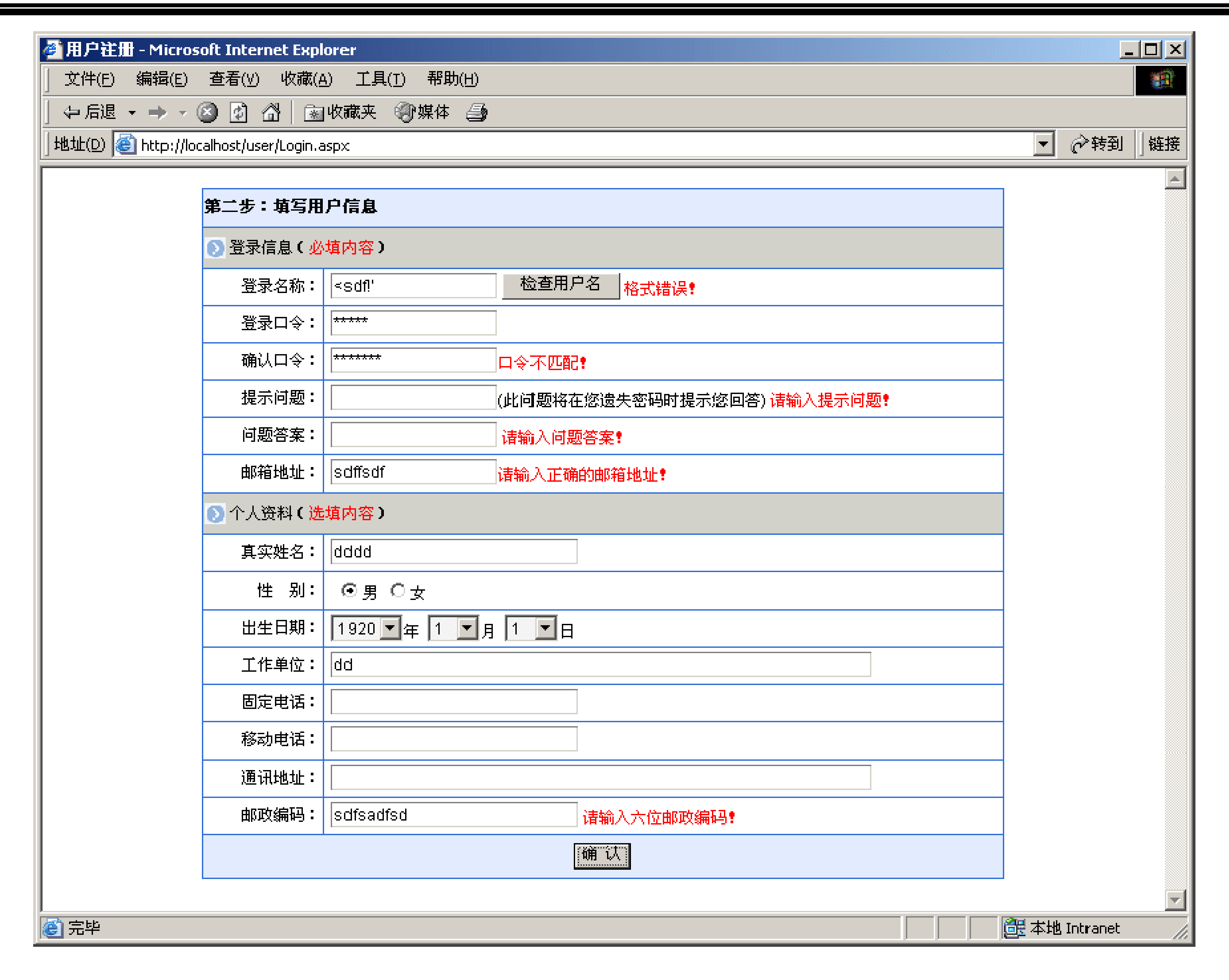


图 8.30　页面预览——输入错误

8.2.3　确认信息

接下来，介绍用户注册中的第 3 步操作，即确认信息。在此步操作中，主要是显示用户所填写的个人资料以供用户确认。

在 ID 为 Step3 的 Panel 控件中，添加一个表格，并插入各种相关的文本标签控件，如图 8.31 所示。

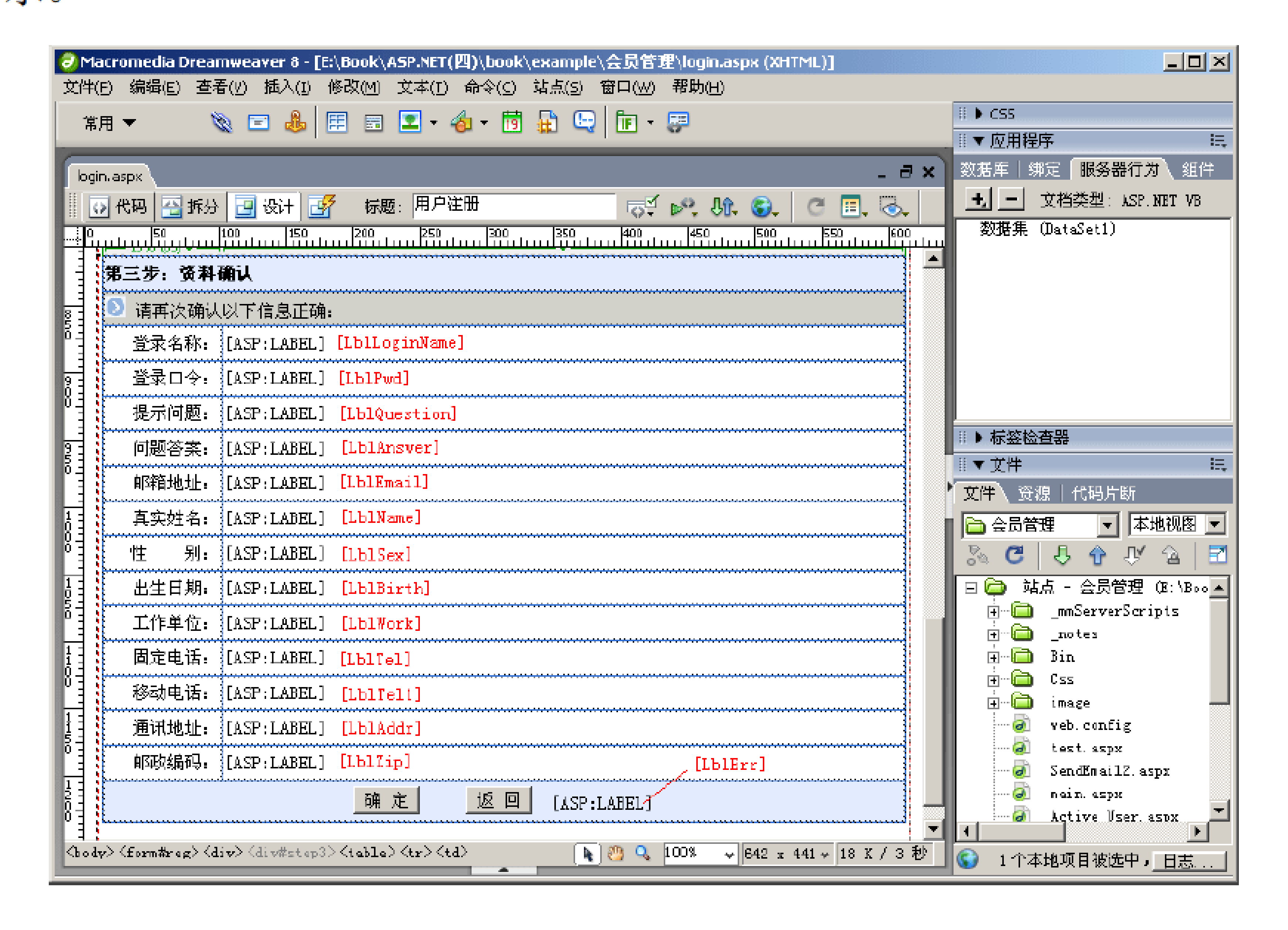


图 8.31　确认信息功能界面

在图中，用文字标记了各个标签控件所对应的 ID 名称。对于除了 ID 为 LblErr 的标签以外的各个标签控件的显示文本，已经在上一步操作中的【确认】按钮的单击事件中进行了赋值。因此，这里只需要进行显示即可。

标签控件 LblErr 主要用于显示错误信息。当单击【确定】按钮时，需要对用户输入的登录名称进行判断。如果该名称在数据库中已经存在，则可在标签 LblErr 中显示相关信息，提示无法保存，并取消对数据库的插入操作。

这里的【确定】按钮也是整个页面中惟一与数据库进行交互操作的按钮，其操作主要分为两步。第一步将用户所填写的个人资料信息保存至数据库中；第二步发送电子邮件至用户注册时所指定的邮箱地址中。

下面来看一下【确定】按钮的单击事件，其代码如下：

【示例代码】

```
Sub Sure2_Click(Sender As Object,E As EventArgs)
  '定义数据库连接
  Dim Cnn As OleDbConnection
  Dim Cmd As OleDbCommand
  Dim Dr As OleDbDataReader
  Dim StrCnn As String
  Dim StrSql As String
  Dim Str1 As String
  '获取数据库的连接字符串
  StrCnn = System.Configuration.ConfigurationSettings.AppSettings("
  MM_CONNECTION_STRING_Cnn")
  '判断登录名称是否重复
  StrSql="Select * from usersheet where loginname='" + thename.text.tostring
  + "'"
  Cnn=new OleDBConnection(StrCnn)
  Cnn.open()
  Cmd = New OleDbCommand(StrSql, Cnn)
  Dr=Cmd.ExecuteReader
  if Dr.read() then
    '存在相同的登录名称，提示信息，且不进行任何操作
    lblerr.text="该用户名已被占用，数据无法保存！"
    Cnn.close()
  else
    Dr.close()
    '不存在相同的登录名称，保存数据，并发送电子邮件
    '定义插入数据的 SQL 语句
    StrSql="insert into usersheet(loginname,UserName,UserPass,Birth,Sex,
    PassQuestion,PassAnswer,Address,Tel1,Tel2,Works,ZipCode,Email)"
    StrSql=StrSql + " Values("
    StrSql = StrSql + "'" + theName.Text.ToString + "',"
    StrSql=StrSql + "'" + username.text.tostring + "',"
    StrSql=StrSql + "'" + LblPwd.text + "',"
    StrSql=StrSql + "'" + cdate(dyear.selecteditem.text.tostring+"-" +
    dmonth.selecteditem.text.tostring + "-" +
    dday.selecteditem.text.tostring) + "',"
    StrSql=StrSql + "'" + sex.selecteditem.text.tostring + "',"
```

```
    StrSql=StrSql + "'" + question.text.tostring + "',"
    StrSql=StrSql + "'" + answer.text.tostring + "',"
    StrSql=StrSql + "'" + addr.text.tostring + "',"
    StrSql=StrSql + "'" + tel1.text.tostring + "',"
    StrSql=StrSql + "'" + tel2.text.tostring + "',"
    StrSql=StrSql + "'" + job.text.tostring + "',"
    StrSql=StrSql + "'" + zip.text.tostring + "',"
    StrSql=StrSql + "'" + email.text.tostring + "')"
    Cmd=new OleDBCommand(strSql,cnn)
    '执行插入数据操作
    Cmd.ExecuteNonQuery()
    Cnn.Close()
    '定义 SmtpClient 类，该类用来实现将电子邮件发送到 SMTP 服务器
    Dim Client As System.Net.Mail.SmtpClient = New
SmtpClient("smtp.sina.com.cn")
    '将 UseDefaultCredentials 属性设置为 False，该属性表示是否发送系统凭据
    Client.UseDefaultCredentials = False
    '设置用于验证发件人身份的凭据，所带参数中前者为发件人的邮箱，后者为邮箱的密码
    Client.Credentials = New System.Net.NetworkCredential(
    "hyh_lxt@sina.com", "hlyxht")
    '指定如何处理待发的电子邮件
    Client.DeliveryMethod = SmtpDeliveryMethod.Network
    '定义 MailMessage 类的实例，该实例可定义使用 SmtpClient 类进行传输的电子邮件
    Dim Message As System.Net.Mail.MailMessage = New MailMessage
    '设置发送至的邮箱地址
    Dim ToAddr As New System.Net.Mail.MailAddress(email.text.tostring)
    '设置发送邮件的邮箱地址
    Dim FromAddr As New System.Net.Mail.MailAddress("hyh_lxt@sina.com")
    Message.To.Add(ToAddr)
    Message.From = FromAddr
    '设置邮件主题
    Message.Subject = "账户激活邮件"
    '设置邮件内容
    Str1 = "<div><b>亲爱的" & UserName.Text & ": </b></div>"
    Str1 = Str1 & "<div style='padding-left:30px;padding-top:30px;'>本站点
    已经收到您的注册信息，请激活以完成您的账户注册。</div>"
    Str1 = Str1 & "<div style='padding-left:60px;padding-top:15px;'><a
    href='http://localhost/user/active_user.aspx?"
    Str1 = Str1 & "name=" & theName.Text & "'>请单击这里以激活您的账户</a></div>"
    Str1 = Str1 & "<div style='padding-left:30px;padding-top:30px;'><hr>
<small><font color=red>此邮件由本站点系统发出，请勿直接回复！</font></small>
</div>"
    Message.Body = Str1
    '设置邮件的编码方式
    Message.BodyEncoding = ASCIIEncoding.GetEncoding("gb2312")
    Message.SubjectEncoding = ASCIIEncoding.GetEncoding("gb2312")
    Message.IsBodyHtml = True
    '发送电子邮件
    Client.Send(Message)
    step1.Visible = False  '隐藏第 1 步操作界面
    step2.visible=false  '隐藏第 2 步操作界面
```

```
    step3.visible=false  '隐藏第 3 步操作界面
    step4.Visible = True  '显示第 4 步操作界面
  end if
End Sub
```

在此段代码中，必须注意以下几点。

① 在定义插入数据的 SQL 语句中，所插入的值大部分都是从第 2 步操作(即填写个人资料)添加各个文本框或下拉列表框中获得，但登录口令一项例外，它是从确认信息中用于显示登录口令的文本标签控件中获得。这是因为，对于密码模式下的文本框控件，一旦数据回发至服务器之后(比如说在单击“填写个人资料”步骤中的【确认】按钮之后)，其文本框中的数据便会清空，而不会保留(这也是出于一种安全考虑)。也就是说，在单击本步骤中的【确定】按钮时，登录口令对应的文本框 thePass 的值已经被清空，因此只能从确认信息中表示登录口令的标签控件 LblPwd 中获取用户所输入的登录口令。

② 对于电子邮件的自动发送，采用了在 ASP.NET 2.0 中新增的 SmtpClient 类。该类允许应用程序使用 Smtp 协议(简单邮件传输协议)来发送电子邮件。与 ASP.NET 1.0 中的 SmtpMail 类相比，该类提供了更为简单、快捷的方法来发送电子邮件。对于 SmtpClient 类的使用，这里不再做详细的叙述，其具体属性和方法的含义可参考相关的 MSDN 文档。

③ 为了确保代码的正常运行，还必须在页首导入以上代码所用到的命名空间，包括 System.Data、System.Data.OleDb 和 System.Net.Mail，这是千万不可忽略的。其中，前两个提供在代码中与数据库进行连接所必需的类，后者则是提供发送电子邮件所必需的类，如图 8.32 所示。

```
<MM:PageBind runat="server" PostBackBind="true" />
<%@ Import NameSpace="System.Data" %>
<%@ Import NameSpace="System.Data.OleDb" %>
<%@ Import Namespace="System.Net.Mail" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
```

导入命名空间

图 8.32　导入命名空间

【返回】按钮的作用是，当用户在确认信息中发现自己的输入有误时，可通过此按钮返回前一页面进行数据的修改。单击【返回】按钮，将触发事件 Return_Click，其代码如下：

【示例代码】

```
Sub Return_Click(Sender As Object,E As EventArgs)
  step1.visible = False  '隐藏第 1 步操作界面
  step2.Visible = True   '显示第 2 步操作界面
  step3.Visible = False  '隐藏第 3 步操作界面
  step4.visible = False  '隐藏第 4 步操作界面
End Sub
```

至此，用户注册的第 3 步操作设计完成，预览界面如图 8.33 所示。

在页面预览中，可能还会存在一个小问题。那就是，前面所输入的中文可能无法正常显示，而是显示为“????”，这是由于 ASP.NET 不支持编码 gb2312 传输所导致的。解决办法很简单：切换至【代码】视图，将页首第一句话中的“ResponseEncoding="gb2312"”去掉即可，如图 8.34 所示。

图 8.33　页面预览

```
<%@ Page Language="VB" ContentType="text/html" ResponseEncoding="gb2312" %>
<%@ Register TagPrefix="MM" Namespace="DreamweaverCtrls" Assembly=
"DreamweaverCtrls,version=1.0.0.0,publicKeyToken=836f606ede05d46a,culture=neutral" %>
```

删除

图 8.34　修改代码

8.2.4　注册成功

接下来介绍用户注册中的第 4 步操作，即注册成功。事实上，在此步操作之前，所有的功能操作均已完成，这里仅仅是显示注册成功的相关信息。

在 ID 为 Step4 的 Panel 控件中，添加一个表格，并插入注册成功的相关提示信息，如图 8.35 所示。

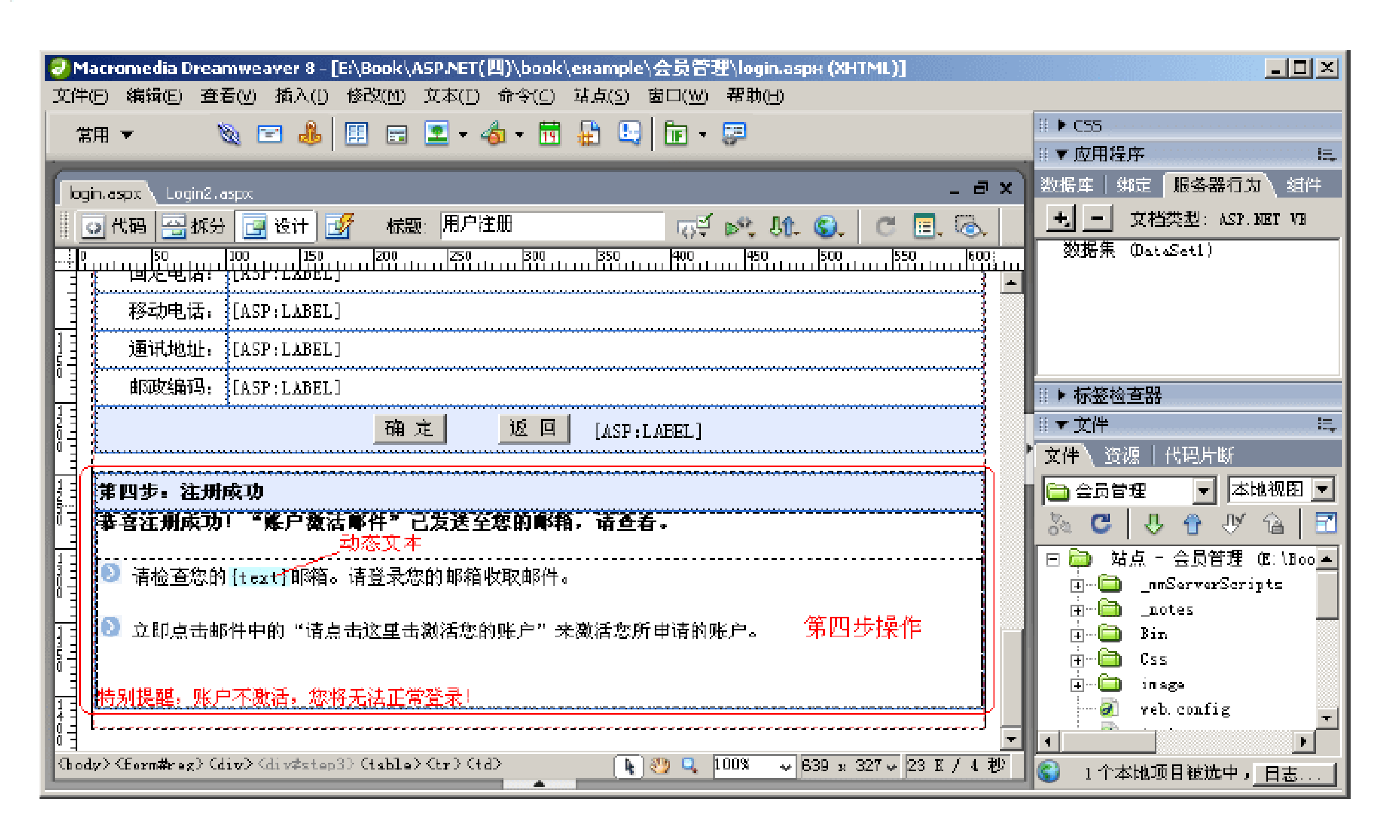


图 8.35　注册成功界面设计

在注册成功信息中，将提示用户“账户激活邮件”已经发送至其注册的邮箱中，并提醒用户必须收取邮件并单击邮件中的链接来激活账户，否则将无法正常登录。这里，通过一个动态文本<%=email.Text %>来获取并显示用户注册时所填写的邮箱地址。

此步操作的页面预览如图 8.36 所示。

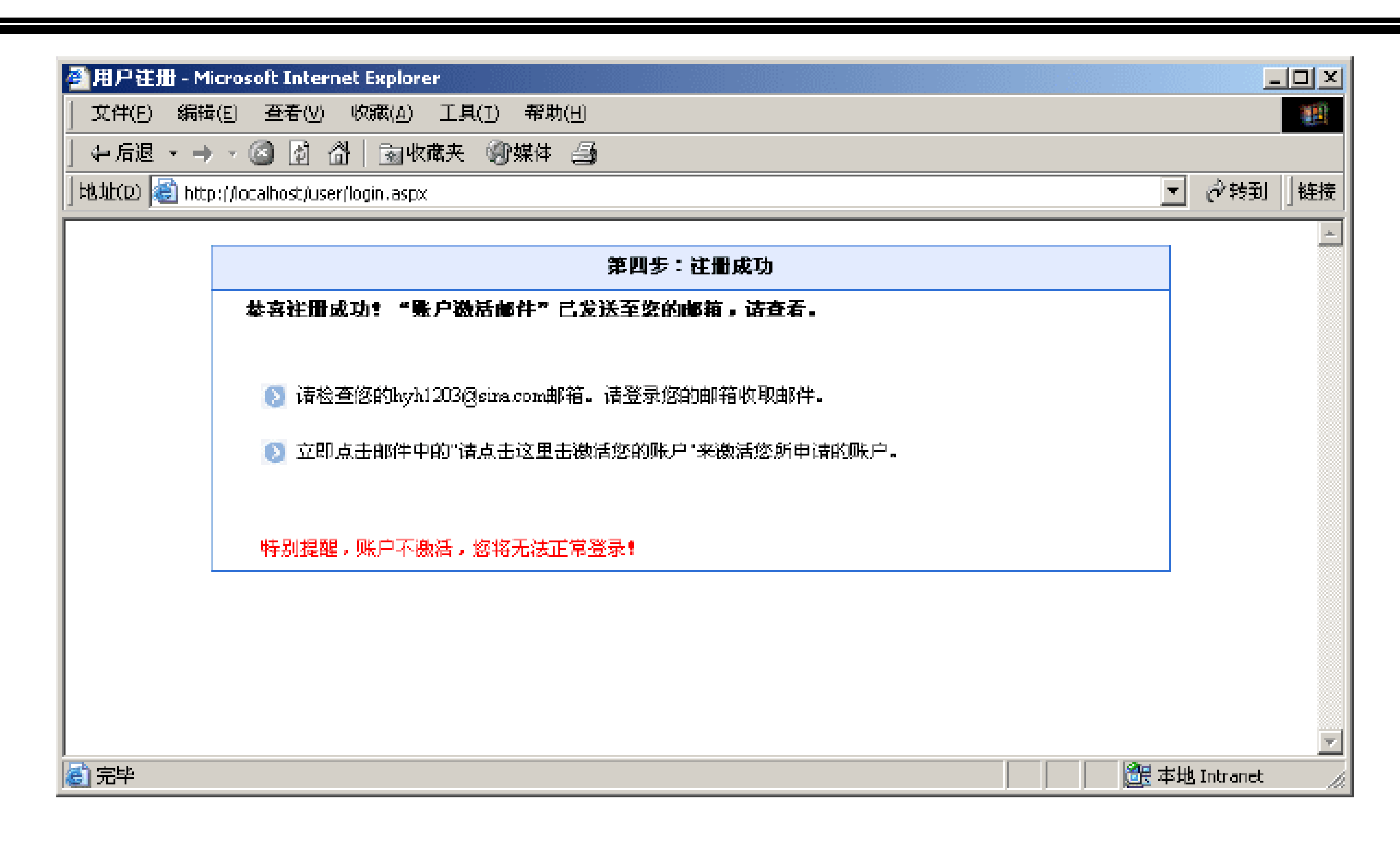


图 8.36　页面预览

至此，Login.aspx 页面全部制作完成。而用户注册的最后一步操作——激活账户将通过页面 Active_User.aspx 来完成。

8.2.5　激活账户

激活账户是用户注册功能中的最后一步操作，其主要目的是验证用户邮箱的正确性，以防恶意注册。

所谓激活账户，其原理是将数据库中指定用户所对应的字段 JHTag 的值设置为 1。因此，页面设计的基本思路是：根据页面传递的参数更新数据库，然后显示相应的提示信息，表示账户已经激活，并提供登录站点的链接按钮。

打开用户注册时所填写的邮箱，可查收到前面所发送的主题为“账户激活邮件”的电子邮件，如图 8.37 所示。

图 8.37　收到的电子邮件

在此邮件中，提供了用于激活账户的链接，其链接地址为：http://localhost/user/active_user.aspx?name=hyh。页面 Active_User.aspx 用于实现激活账户的功能，其所带参数 name 表示所要激活账户的会员的登录名称。一般来说，在正式应用中，所传递的参数值(如这里的 hyh)是需要进行加密的，而不会采用明文方式。这里仅介绍功能的实现，因此没有考虑加密的问题。该激活页面的具体实现如下。

1. 设计页面的基本框架

为了防止用户恶意激活不存在的账户，可以添加一个容器控件 PlaceHolder，将要显示的信息包含在其中。当指定的账户不存在时，则隐藏该容器，包括该容器内的所有文本或控件，此时页面将无任何显示。当指定的账户存在并执行了激活操作时，页面上的提示信息才真正显示出来。

(1) 单击【ASP.NET-插入】工具栏中的按钮，在弹出的【标签选择器】对话框中选择【ASP.NET 标签】|【Web 服务器】分类，并在右边的列表框中选择 asp:PlaceHolder。单击【插入】按钮，在弹出的【标签编辑器】对话框中，设置控件 ID 为 PlaceHolder1，并取消【可见】复选框的选择，如图 8.38 所示。

图 8.38　添加 PlaceHolder 控件

(2) 将光标置于控件 PlaceHolder1 之内，并在其中添加所要显示的相关文本信息，如图 8.39 所示。

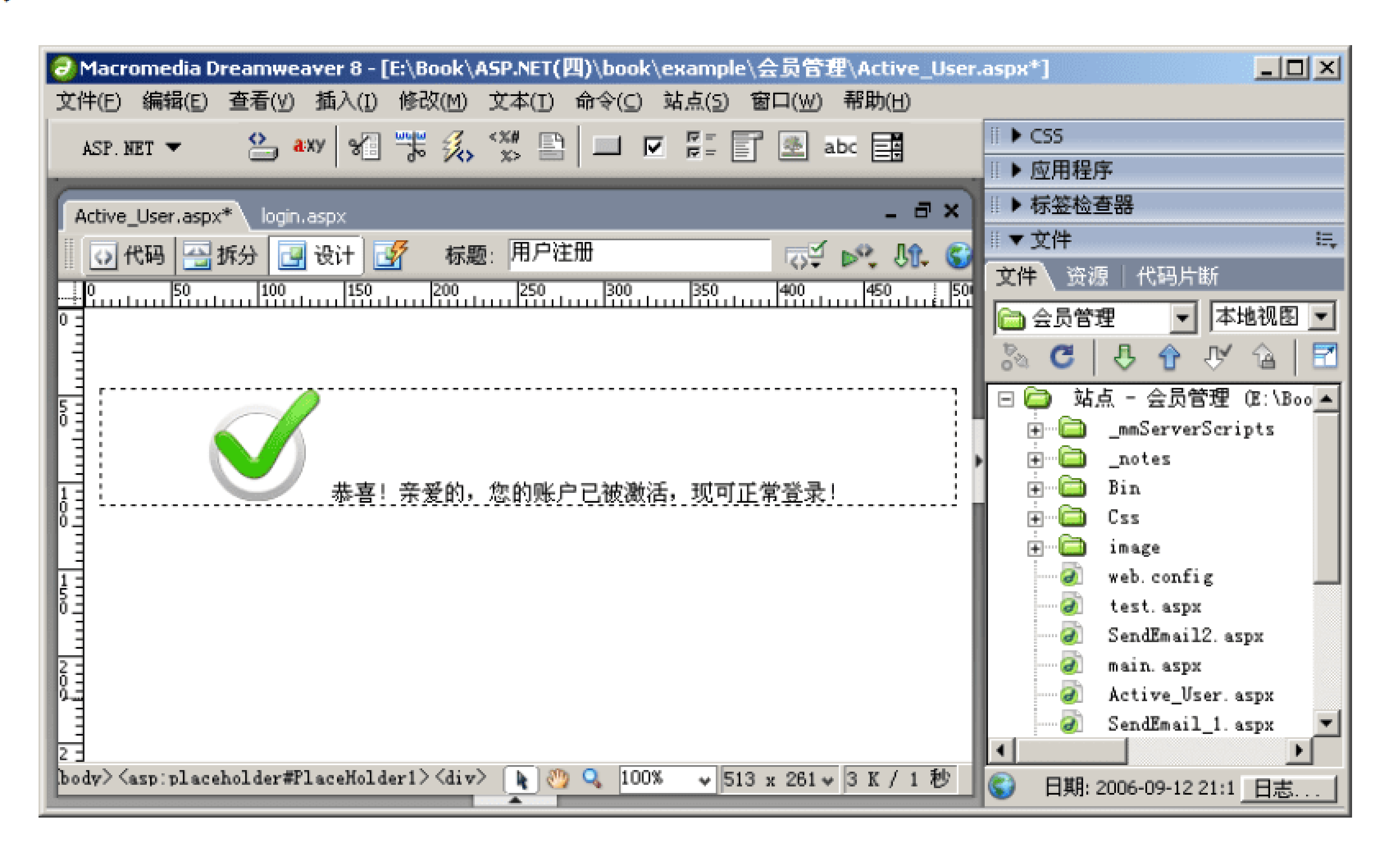


图 8.39　添加文本

(3) 为了在文本“亲爱的”后面显示激活用户的真实名称，这里需要创建一个数据集。打开【应用程序】面板组，切换至【服务器行为】面板，单击按钮，在弹出的菜单中选择【数据集】命令，将弹出【数据集】对话框，如图 8.40 所示。

(4) 设置数据集的名称为 DataSet1，选择连接 Cnn，选择数据表格 UserSheet，在筛选条件中设置为字段 LoginName 等于 URL 参数 Name。单击【确定】按钮，完成设置。

图 8.40　添加数据集

(5)　在【应用程序】面板组中，将选项切换至【绑定】面板，展开刚才创建的数据集 DataSet1，如图 8.41 所示。

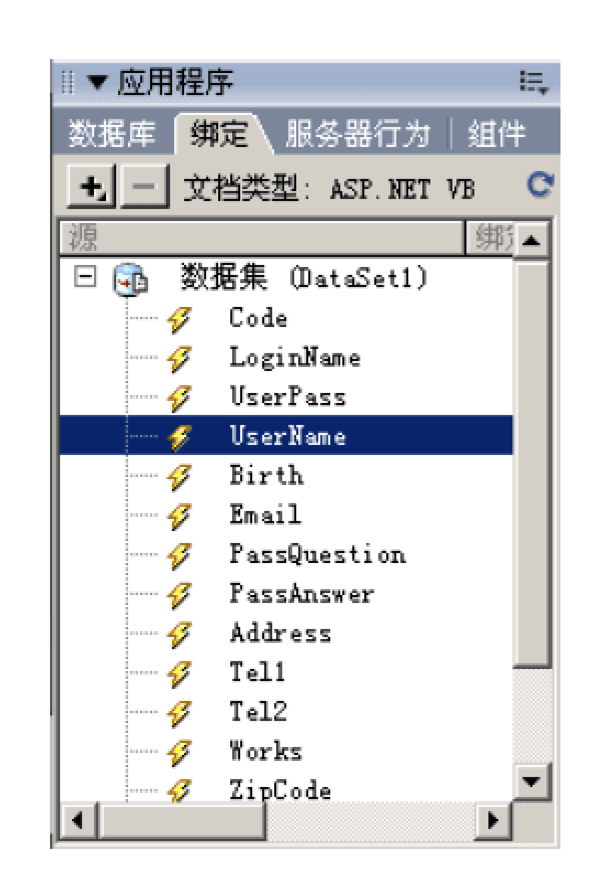


图 8.41　【绑定】面板

选择字段 UserName，并将其拖至设计窗口中的文本“亲爱的”后面。这样，创建了一个绑定文本，该文本将显示当前激活用户的真实名称。

2. 创建一个指向登录页面的链接

接下来，在提示文本的下方创建一个指向登录页面的链接。

(1)　单击【ASP.NET-插入】工具栏中的按钮，在弹出的【标签选择器】对话框中，选择【ASP.NET 标签】|【Web 服务器】分类，并在右边的列表框中选择 asp:HyperLink。

(2)　单击【插入】按钮，在弹出的【标签编辑器】对话框中，设置该 HyperLink 控件的 ID 为 Link1，导航 URL 为 Default.aspx (即登录页面)，目标选择_self(即在本窗口中打开新链接)，文本设置为“登录站点”，如图 8.42 所示。

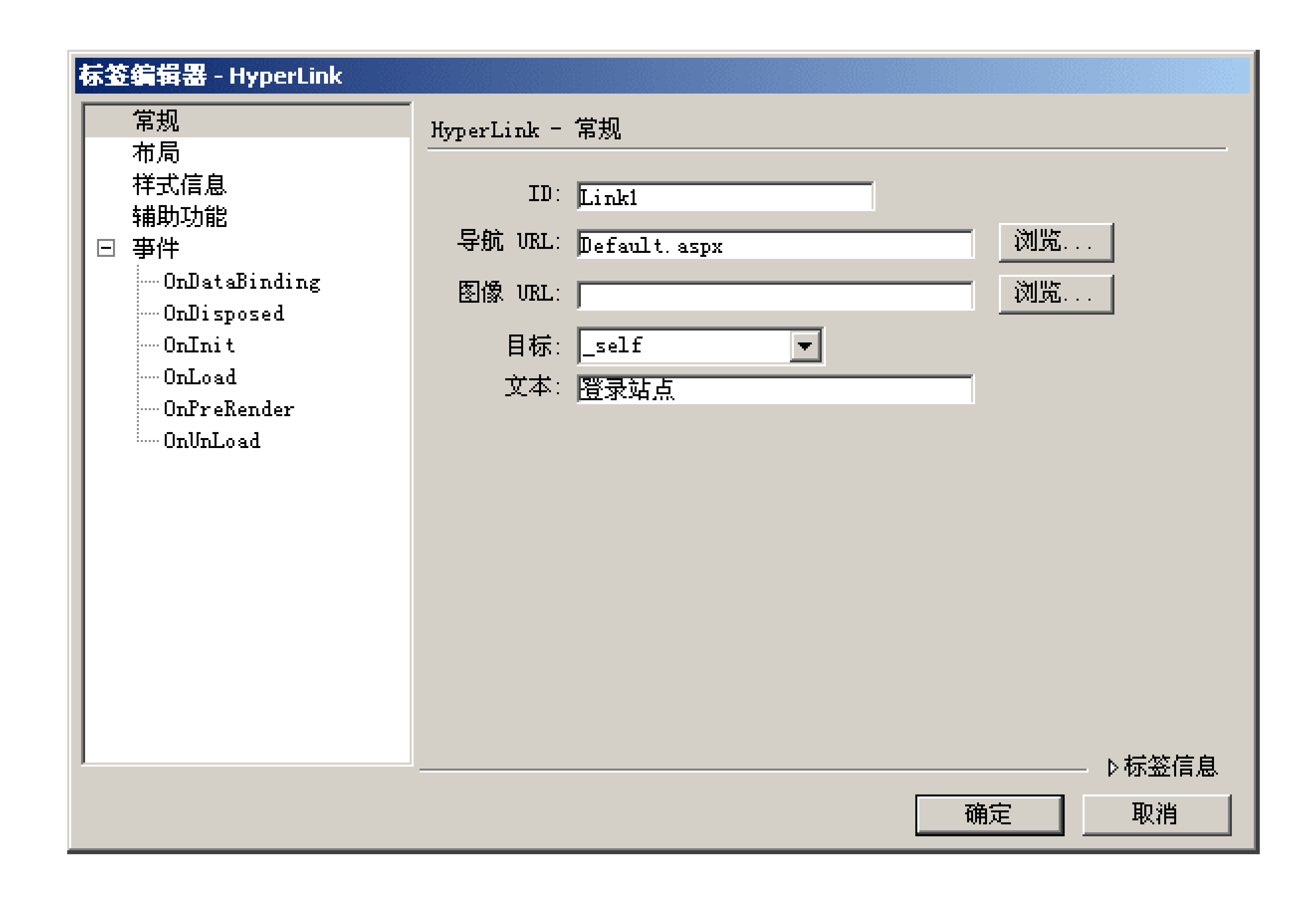


图 8.42　添加 HyperLink 控件

(3)　在【布局】分类中，设置其前景颜色为红色；并在【样式信息】分类中，设置粗体显示。

下面，切换至【代码】视图，在 Page_Load 事件中添加以下代码：

【示例代码】

```
Sub Page_Load(ByVal Sender As Object, ByVal E As EventArgs)
  Dim Cnn As OleDbConnection
  Dim Cmd As OleDbCommand
  Dim StrCnn As String
  Dim StrSql As String
  If (Not Page.IsPostBack) Then
    '获取数据集 DataSet1 的记录数，判断要激活的用户是否存在
    If DataSet1.defaultView.Table.Rows.Count>0 Then
      '存在要激活的用户，执行更新操作，并显示 PlaceHolder 控件
      StrCnn = System.Configuration.ConfigurationSettings.AppSettings("
      MM_CONNECTION_STRING_Cnn")
      Cnn = New OleDbConnection(StrCnn)
      Cnn.Open()
      '执行更新语句
      StrSql = "update usersheet set jhtag='1' where loginname='" &
      Trim(Request.QueryString("name")) & "'"
      Cmd = New OleDbCommand(StrSql, Cnn)
      Cmd.ExecuteNonQuery()
      Cnn.close()
      '显示 PlaceHolder 控件
      PlaceHolder1.visible=true
    else
      '不存在要激活的用户，隐藏 PlaceHolder 控件
    PlaceHolder1.visible=false
    End If
  End If
End Sub
```

根据数据集 DataSet1 所筛选的结果，可以得知当前要激活的用户是否存在。如果存在，则执行激活操作，并显示相关信息。如果不存在，则隐藏信息的显示。

提示：此页面需导入命名空间 System.Data 和 System.Data.OleDb。

至此，用户激活页面的制作全部完成，页面预览效果如图 8.43 所示。

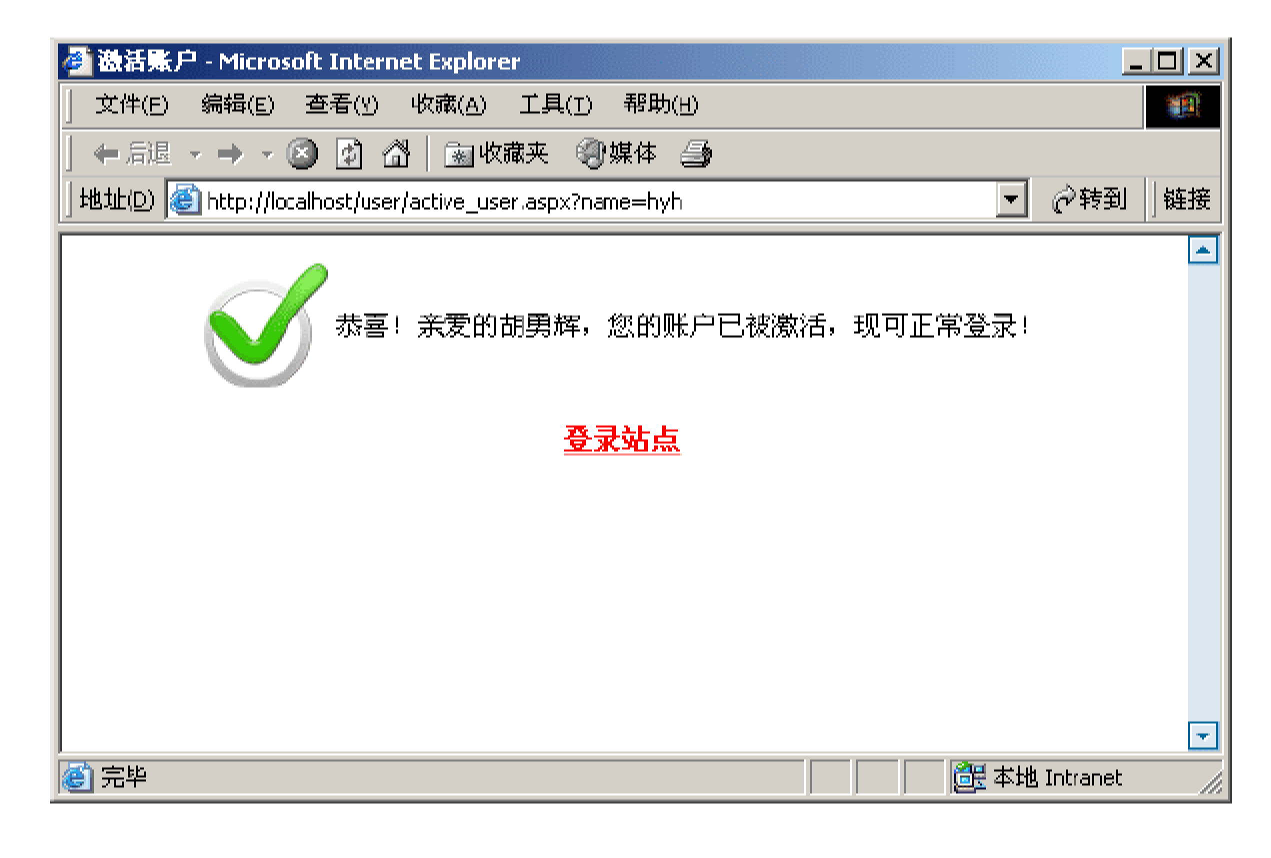


图 8.43　页面预览效果

8.3　用 户 登 录

8.3.1　登录站点

登录站点是会员管理中的一个基本功能，它允许用户通过注册的登录名称和登录密码进入站点。在本系统中，由于提供了账户激活功能，因此在检测用户名和密码是否正确的同时，还需判断该账户是否已被激活；否则，是不能正常登录的。

下面，来看一下登录页面的具体实现。

对于页面的界面设计，这里不做过多的叙述，图 8.44 所示的是该页面的基本界面。

(1) 在界面中插入两个文本框控件 theName 和 thePwd，它们分别对应于登录名称和登录密码的输入。其中，文本框 thePwd 的文本模式为密码。

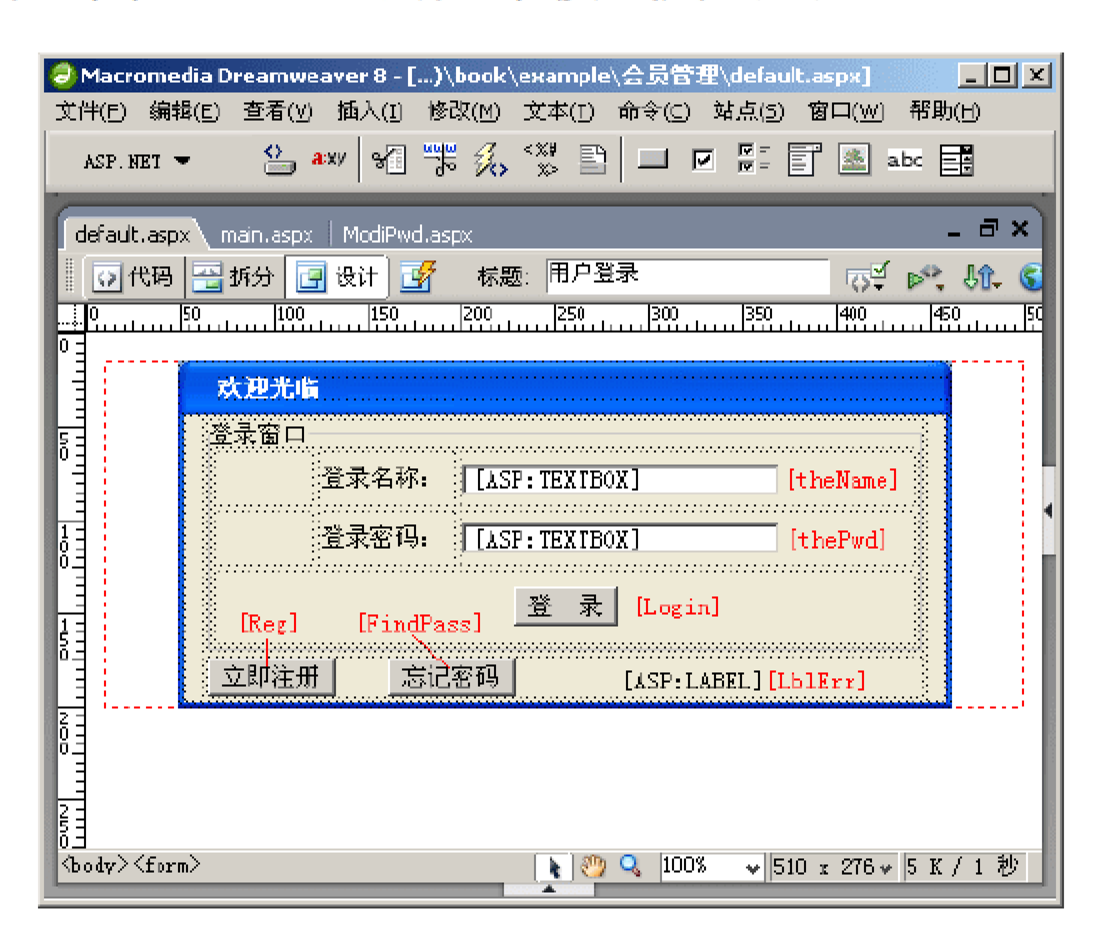


图 8.44　界面设计

(2) 添加三个按钮控件，其文本分别设置为“登录”、“立即注册”和“忘记密码”，对应的 ID 分别为 Login、Reg 和 FindPass，按钮的单击事件分别为 Login_Click、Reg_Click 和 FindPass_Click。

为了显示错误信息，还需要在界面上插入一个标签控件 LblErr。该控件用于显示当登录名称或密码错误，或者此账号尚未被激活时的相关错误信息。

(3) 在登录页面中，登录名称和登录密码是必须输入的。因此，还需要在两个文本框之后分别添加 RequiredFieldValidator 控件(必须输入验证控件)，其属性设置分别如图 8.45 和图 8.46 所示。

验证控件 Require1 用于对登录名称的输入进行验证，而 Require2 则用于对登录密码的输入进行验证。为了避免当用户单击【立即注册】按钮和【忘记密码】按钮时，由于页面验证失败而无法执行正常的操作，这里需要将按钮控件 Reg 和 FindPass 的 CausesValidation 属性(即取消选择标签编辑器中的【原因确认】复选框)设置为 False。

图 8.45　添加 RequiredFieldValidator 控件 Require1

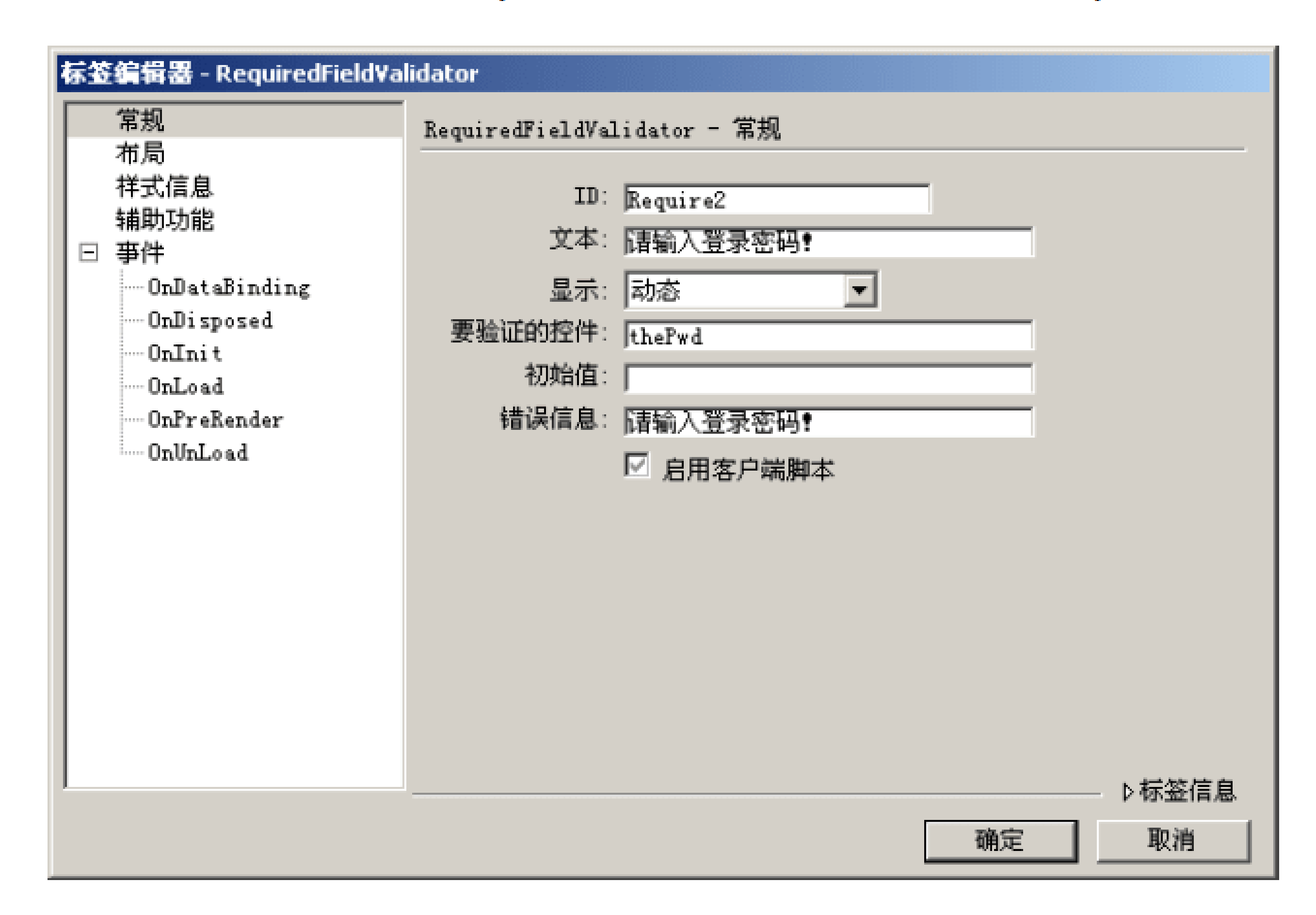


图 8.46　添加 RequiredFieldValidator 控件 Require2

界面设计基本完成，接下来看一下具体功能的实现。为了判断用户的输入是否正确，这里需要创建一个数据集。

(1)　打开【应用程序】面板组，切换至【服务器行为】面板，单击 按钮，在弹出的菜单中选择【数据集】命令，此时将弹出【数据集】对话框，如图 8.47 所示。

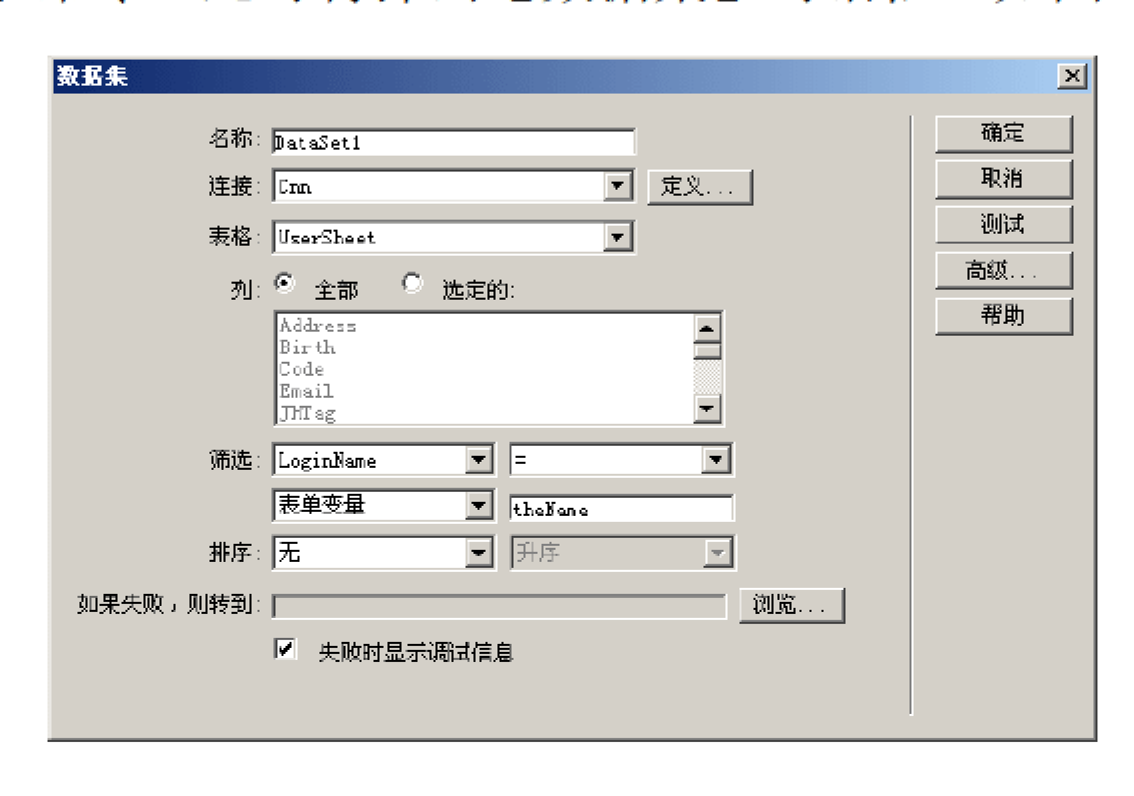


图 8.47　【数据集】对话框

(2)　设置数据集名称为 DataSet1，数据库连接选择 Cnn，在表格下拉列表框中选择

UserSheet，同时设置筛选条件为字段 LoginName 等于表单变量 theName。同时，还需要添加一个筛选条件，以判断密码的正确性。

(3)　单击【高级】按钮，在弹出的对话框中单击参数右侧的 + 按钮，添加一个新的参数。此时，将弹出【添加参数】对话框，如图 8.48 所示。

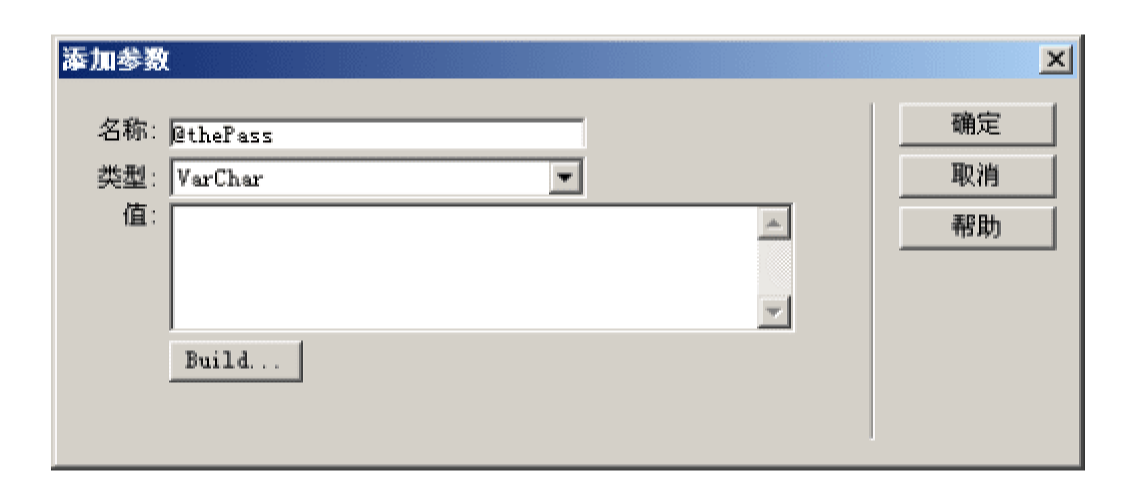


图 8.48　【添加参数】对话框

(4)　输入参数名称“@thePass”，在参数类型中选择 VarChar，参数的值可通过单击 Build 按钮来创建，如图 8.49 所示。

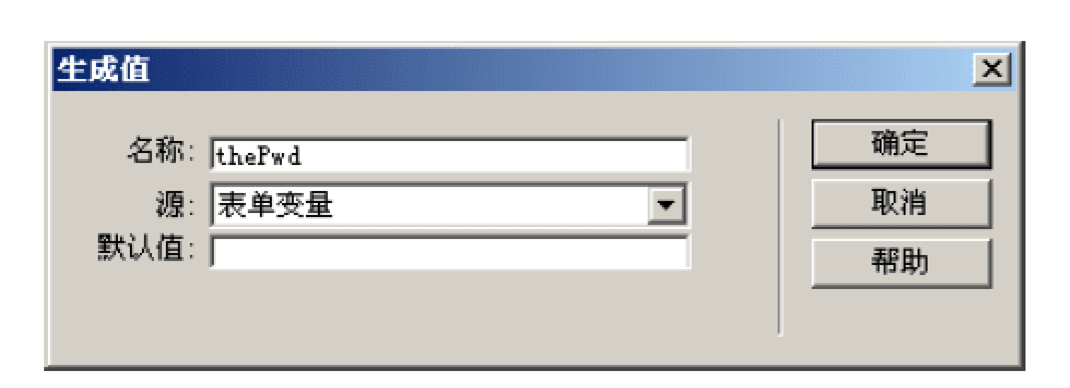


图 8.49　【生成值】对话框

(5)　单击【确定】按钮，完成参数的创建。返回【数据集】对话框的高级模式，在所显示的 SQL 文本框中添加一个条件表达式以判断密码是否正确，如图 8.50 所示。

单击【确定】按钮，完成数据集的创建。当用户输入登录名称和登录密码并单击【登录】按钮时，根据数据集中的记录是否为空，可以判断用户所输入的用户名及密码是否正确。

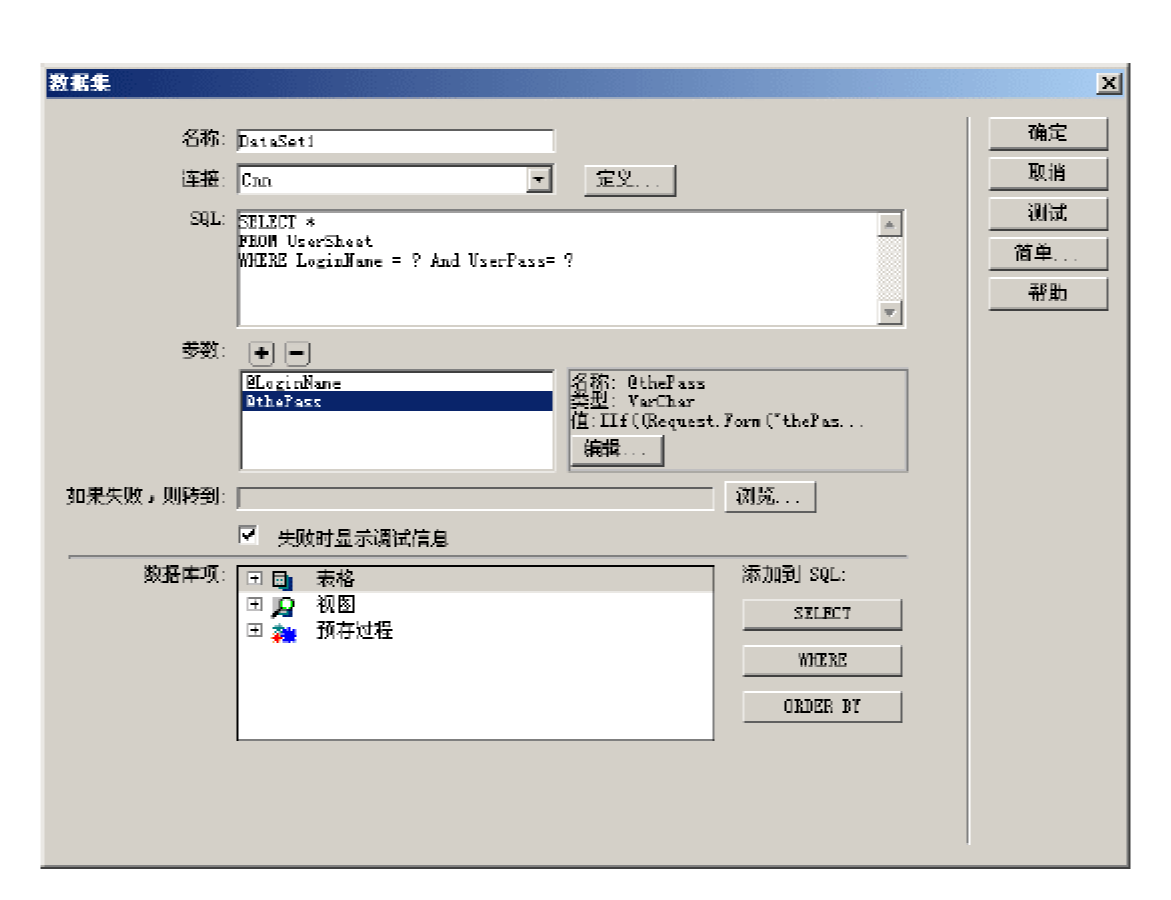


图 8.50　【数据集】对话框的高级模式

(6)　切换至【代码】视图，添加各个按钮的单击事件，代码如下：

【示例代码】

```
<script runat="server">
```

```
'添加按钮“登录”的单击事件
Sub Login_Click(ByVal sender As Object, ByVal E As EventArgs)
  '定义变量 JHStr，用于存储账户是否激活标志
  dim JHStr as String
  '判断数据集 DataSet1 中的记录是否为空
  If DataSet1.DefaultView.Table.Rows.Count>0 then
    '数据集 DataSet1 中存在记录，说明用户名称及密码正确，但仍需要判断其账户是否激活
    '获取数据集 DataSet1 中的第一条记录(一般来说，只可能有一条记录)的字段 JHTag 的值
    JHStr=DataSet1.DefaultView.Table.Rows(0)("JHTag")
    If trim(JHStr)="1" Then
'该账户已经激活，可正常登录
'将其用户编号、用户名称以及当前登录时间存储至 Session 变量中
      Session("userid") = DataSet1.DefaultView.Table.Rows(0)("code")
      Session("username") = DataSet1.DefaultView.Table.Rows(0)("username")
      Session("logintime") = System.DateTime.Now
      '页面跳转至登录后的主页面
      Response.Redirect("main.aspx")
    else
      'JHStr 的值为 0，说明该账户尚未激活，提示错误信息
      LblErr.Text = "此账户尚未激活，无法正常登录！"
    end if
  else
    '数据集 DataSet1 中的记录为空，无筛选结果，提示名称或密码错误
    LblErr.Text = "登录名称或密码错误！"
  end if
End Sub
' 添加【忘记密码】按钮的单击事件
Sub FindPass_Click(ByVal sender As Object, ByVal E As EventArgs)
  '将页面跳转至找回密码页面 LoadPass.aspx
  Response.Redirect("LoadPass.aspx")
End Sub
'添加【立即注册】按钮的单击事件
Sub Reg_Click(ByVal sender As Object, ByVal E As EventArgs)
  '将页面跳转至用户注册页面 Login.aspx
  Response.Redirect("Login.aspx")
End Sub
</script>
```

在【登录】按钮的单击事件中，首先根据数据集 DataSet1 的记录数，判断是否存在与用户所输入的登录名称和登录密码相匹配的用户信息。如果记录数为 0，则表示不存在匹配的信息，提示用户名或密码错误。如果记录数大于 0，则说明用户所输入的登录名称和密码正确；但此时仍需根据匹配记录中的字段 JHTag 的值来判断该账号是否已激活。当 JHTag 的值为 0 时，表示账号尚未被激活。当 JHTag 的值为 1 时，表示账号已被激活，这时用户可以正常登录。为了在后续的页面中验证用户的合法身份，这里创建了三个 Session 变量：UserID(用户编号)、UserName(用户名称)和 LoginTime(登录时间)，并将页面跳转至站点的主页面 Main.aspx。

对于【立即注册】按钮和【忘记密码】按钮的单击事件，其操作均相当简单，直接将页面跳转到相应的操作页面即可。

至此，页面功能设计完成，界面预览如图 8.51 所示。

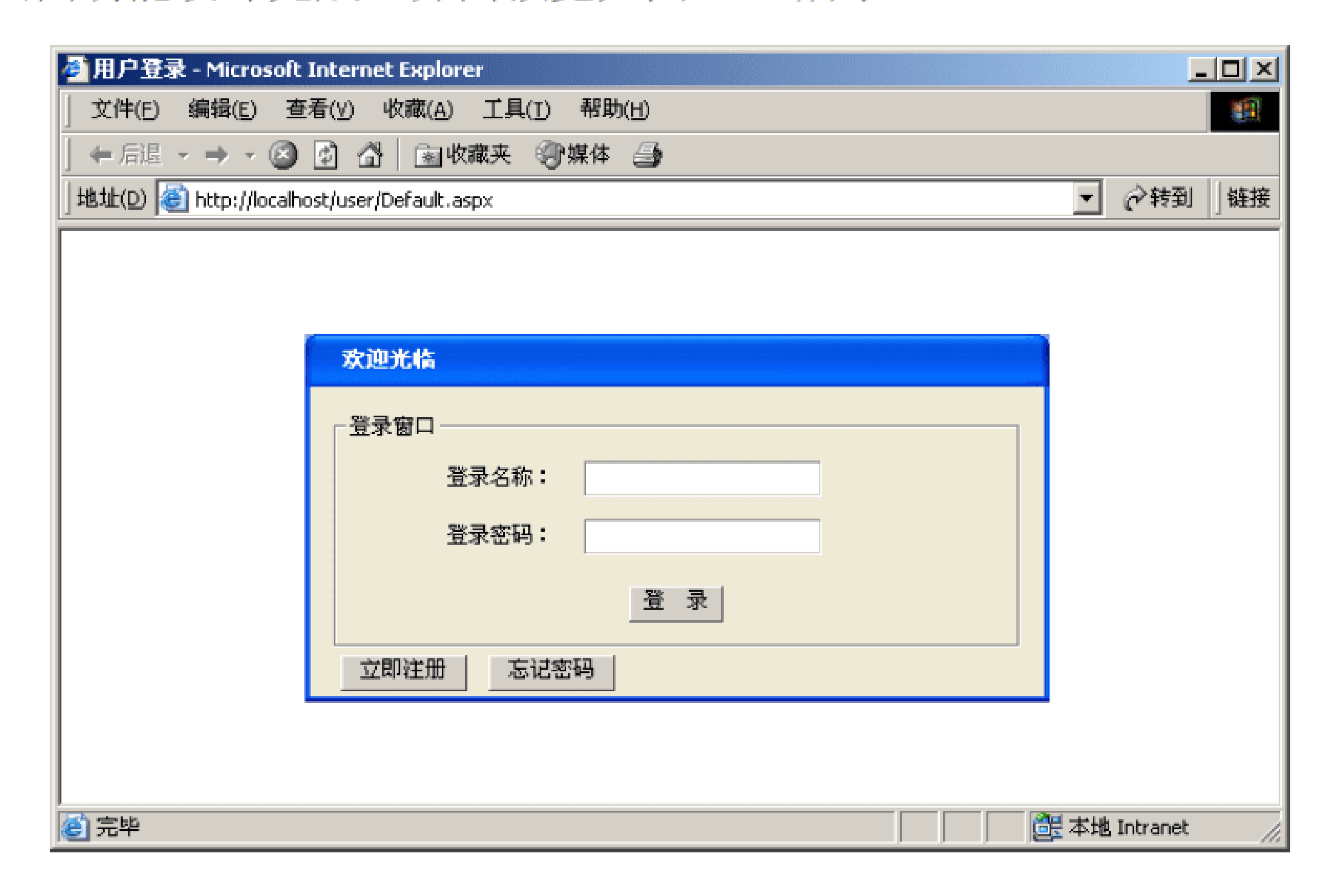


图 8.51　页面预览

8.3.2　主页面

用户登录成功后，即可进入站点的主页面。在实际应用中，不同的站点其主页面的内容是完全不同的。在本例中，主页面仅仅是提示用户已成功登录站点，并提供注销登录的链接和修改个人信息的链接。

此外，有些用户可能在未登录的情况下直接在地址栏中输入主页面的地址来进行访问。此时，出于安全考虑，需要对其进行判断，当用户尚未登录时，应给出相应提示，并提供登录页面的相应链接。

综上考虑，在主页面中需要添加一个标签控件和三个按钮控件，如图 8.52 所示。

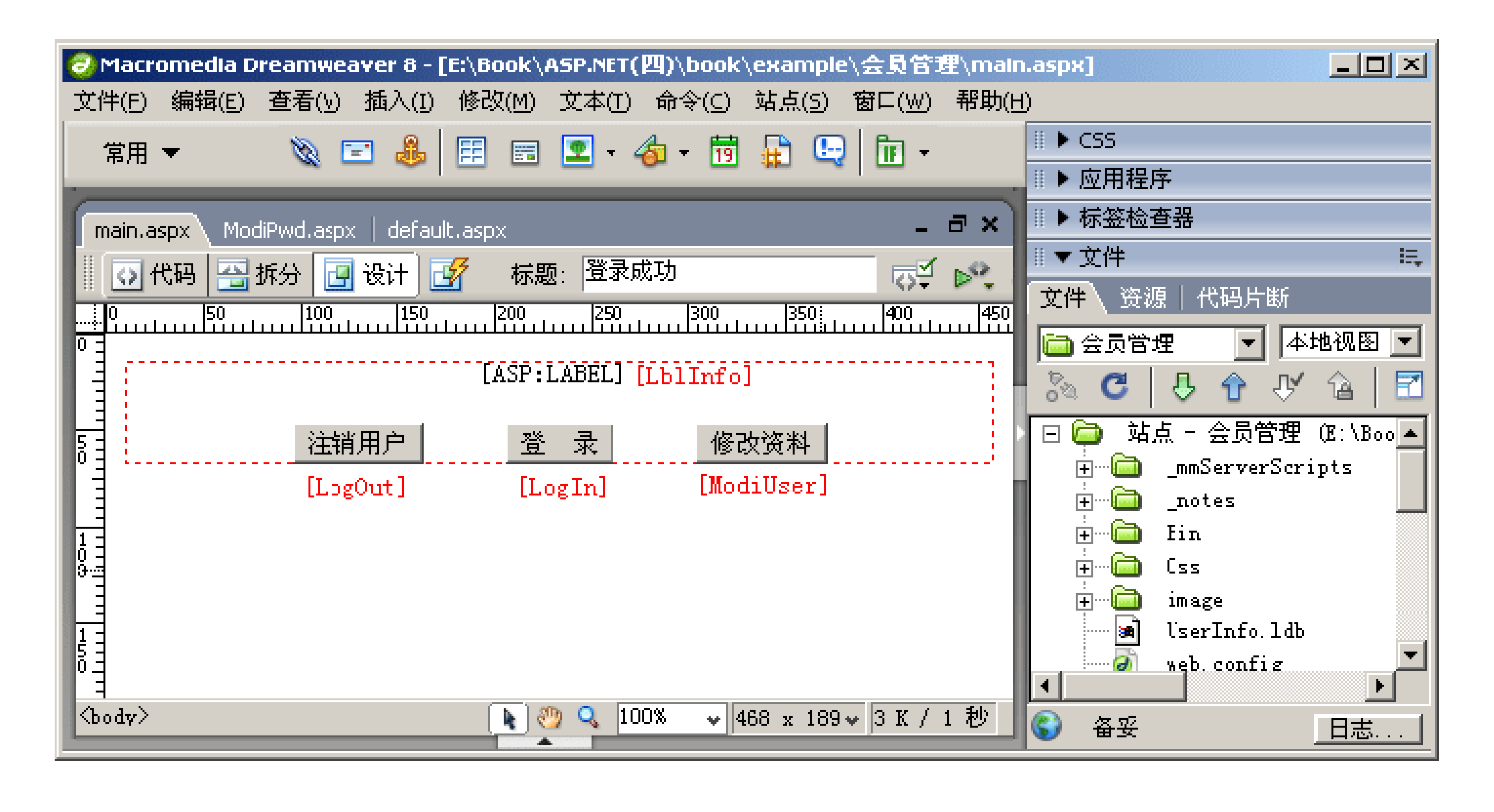


图 8.52　主页面设计

在此页面中，标签控件 LblInfo 用于显示提示信息，LogOut(注销用户)按钮用于执行注销功能，LogIn(登录)按钮用于将页面链接至登录页面，ModiUser(修改资料)按钮则用于将页面链接至修改信息页面。但是，在同一时间，三个按钮不可能同时显示。当用户为合法用户(即通过登录后进入此页面)时，仅显示 LogOut 按钮和 ModiUser 按钮。当用户为非法

用户(即未通过登录而直接访问此页面)时，仅显示 LogIn 按钮。当用户通过单击 LogOut 按钮执行注销功能后，仅显示 LogIn 按钮，且其按钮文本将变为“重新登录”。对于以上三种情况，标签控件 LblInfo 均将显示不同的文本信息。这些，均由代码来实现。

下面，介绍代码的具体实现。

切换至【代码】视图，添加 Page_Load 事件以及各按钮的单击事件，代码如下：

【示例代码】

```
<script runat="server">
'添加页面加载时执行的 Page_Load 事件
Sub Page_Load(ByVal Sender As Object, ByVal e As EventArgs)
  If Not Page.IsPostBack Then
    '判断 Session 变量是否为空，并以此判断用户是否已经登录系统
    If IsDBNull(Session("userid")) Or Trim(Session("userid")) = "" Then
      'Session 变量为空，说明用户为非法用户
      LblInfo.Text = "对不起，您尚未登录系统！"
      LogOut.Visible = False  '隐藏【注销用户】按钮
      ModiUser.Visible = False  '隐藏【修改资料】按钮
      LogIn.Visible = True  '显示【登录】按钮
    Else
'Session 变量不为空，说明用户为合法用户
'通过 Session 变量 username，来获取当前登录的用户名称
      LblInfo.Text = "恭喜，" & Trim(Session("username")) & "，您已成功登录系
      统！"
      LogOut.Visible = True    '显示【注销用户】按钮
      ModiUser.Visible = True  '显示【修改资料】按钮
      LogIn.Visible = False    '隐藏【登录】按钮
    End If
  End If
End Sub
'添加【注销用户】按钮的单击事件
Sub LogOut_Click(ByVal sender As Object, ByVal E As EventArgs)
  '清空 Session 变量
  Session("userid") = ""
  Session("username") = ""
  Session("logintime") = ""
  '提示注销成功
  LblInfo.Text = "用户注销成功，您可安全关闭此页面，也可重新登录！"
  '将【登录】按钮的显示文本改为【重新登录】
  LogIn.Text = "重新登录"
  LogOut.Visible = False    '隐藏【注销用户】按钮
  ModiUser.Visible = False  '隐藏【修改资料】按钮
  LogIn.Visible = True      '显示【重新登录】按钮
End Sub
'添加【登录】按钮的单击事件
Sub Login_Click(ByVal sender As Object, ByVal E As EventArgs)
  '页面跳转至登录页面 Default.aspx
  Response.Redirect("default.aspx")
End Sub
'添加【修改资料】按钮的单击事件
Sub Modi_Click(ByVal sender As Object, ByVal E As EventArgs)
```

```
  '页面跳转至修改信息页面 ModiUser.aspx
  Response.Redirect("ModiUser.aspx")
End Sub
</script>
```

页面预览效果如图 8.53 所示。

图 8.53　页面预览效果

单击【注销用户】按钮，页面效果如图 8.54 所示。

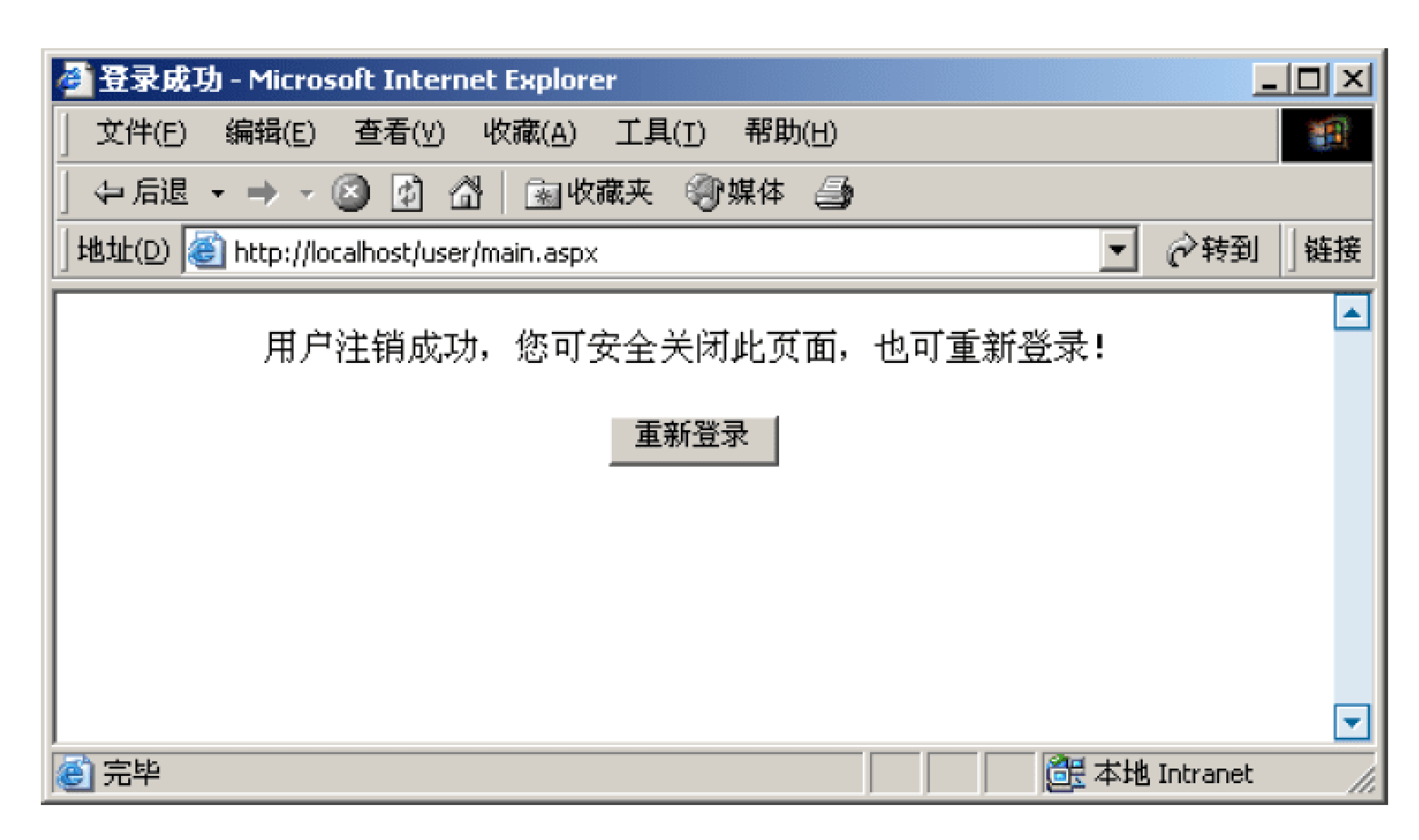


图 8.54　注销后的页面效果

8.4　找 回 密 码

找回密码是会员管理中一个必不可少的功能。对于用户来说，如果长期没有登录系统，很可能会忘记自己账号的密码。此时，则可通过找回密码功能来寻回自己账号的密码。

并不是任何人都可轻易获取账号的密码，首先必须提供正确的账号，然后输入正确的生日并正确回答密码提示问题，才能通过系统获取该账号的密码。其中，生日信息、密码提示问题以及提示问题的答案均是由用户在注册时所设置的。

首先来看页面的设计，如图 8.55 所示。

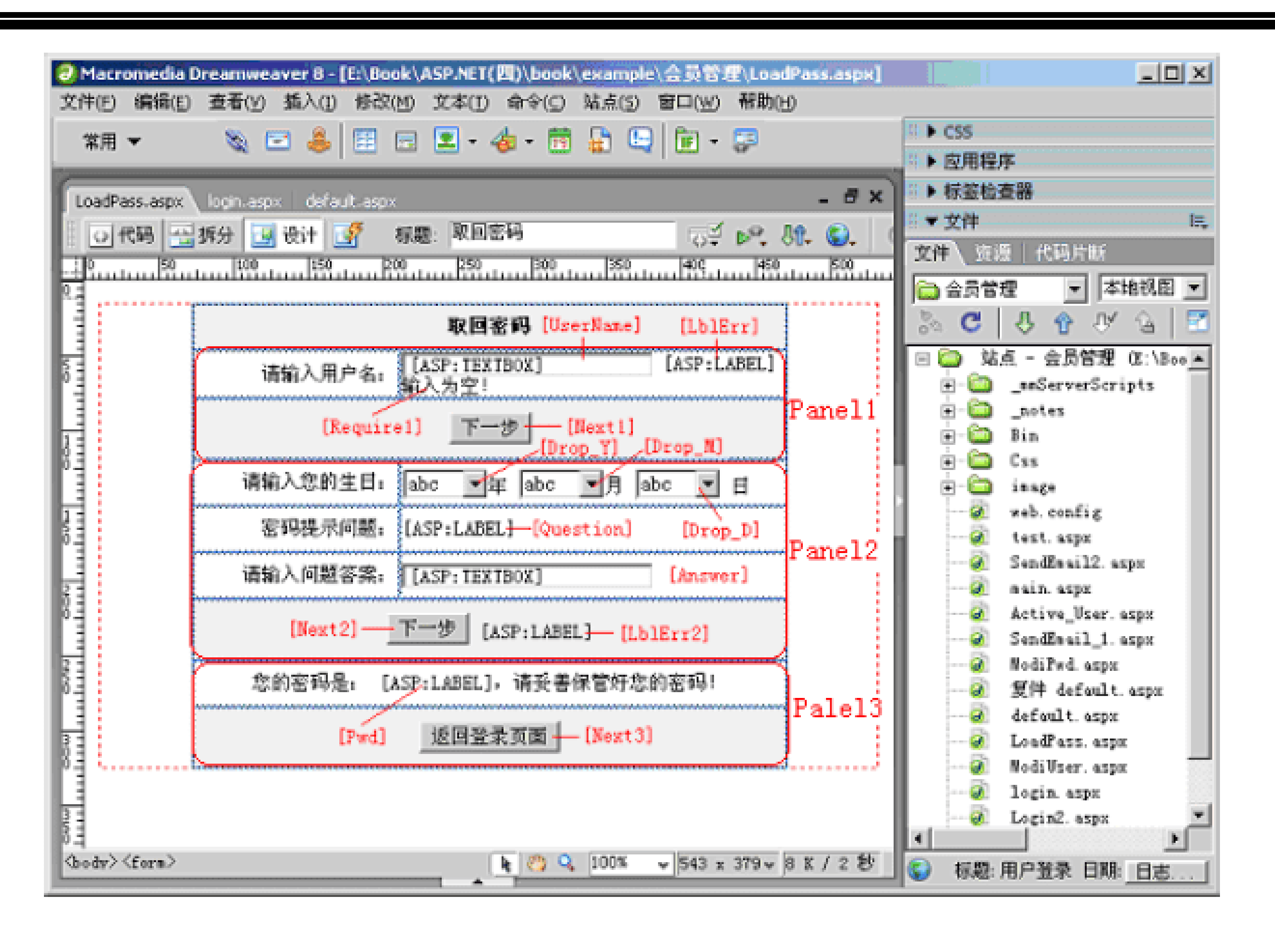


图 8.55　页面设计

在图 8.55 中，用文字标记了各 ASP.NET 控件对应的 ID 名称。页面中，首先添加了三个 Panel 控件，用于控制找回密码的三步操作的界面显示和隐藏。

在控件 Panel1 中，提供了找回密码的第一个步骤的操作界面，即用户名称的输入。在用于输入用户名的文本框控件 UserName 之后，添加了一个标签控件 LblErr 和一个 RequiredFieldValidator 控件 Require1。其中，标签控件 LblErr 用于显示当用户名不存在时的错误信息，而 RequiredFieldValidator 控件 Require1 则用于限制文本框控件的输入值不能为空。控件 Require1 的属性设置如图 8.56 所示。

初始状态下，仅显示 Panel1 控件而隐藏 Panel2 和 Panel3 控件。当输入用户名称并经 Next1 按钮控件中的代码确认该名称存在时，则显示 Panel2 控件，而隐藏 Panel1 控件。

在 Panel2 控件中，提供了找回密码的第二个步骤的操作界面，即出生日期以及密码提示问题答案的输入。在 Next2 按钮控件的一侧，添加了一个 LblErr1 标签控件，该控件用于显示当出生日期或密码提示问题的答案与指定用户名的信息不符时的错误信息。

当用户在 Panel2 控件中输入的出生日期与密码提示问题答案通过 Next2 按钮控件中的代码验证正确时，则显示 Panel3 控件，而隐藏 Panel2 控件。

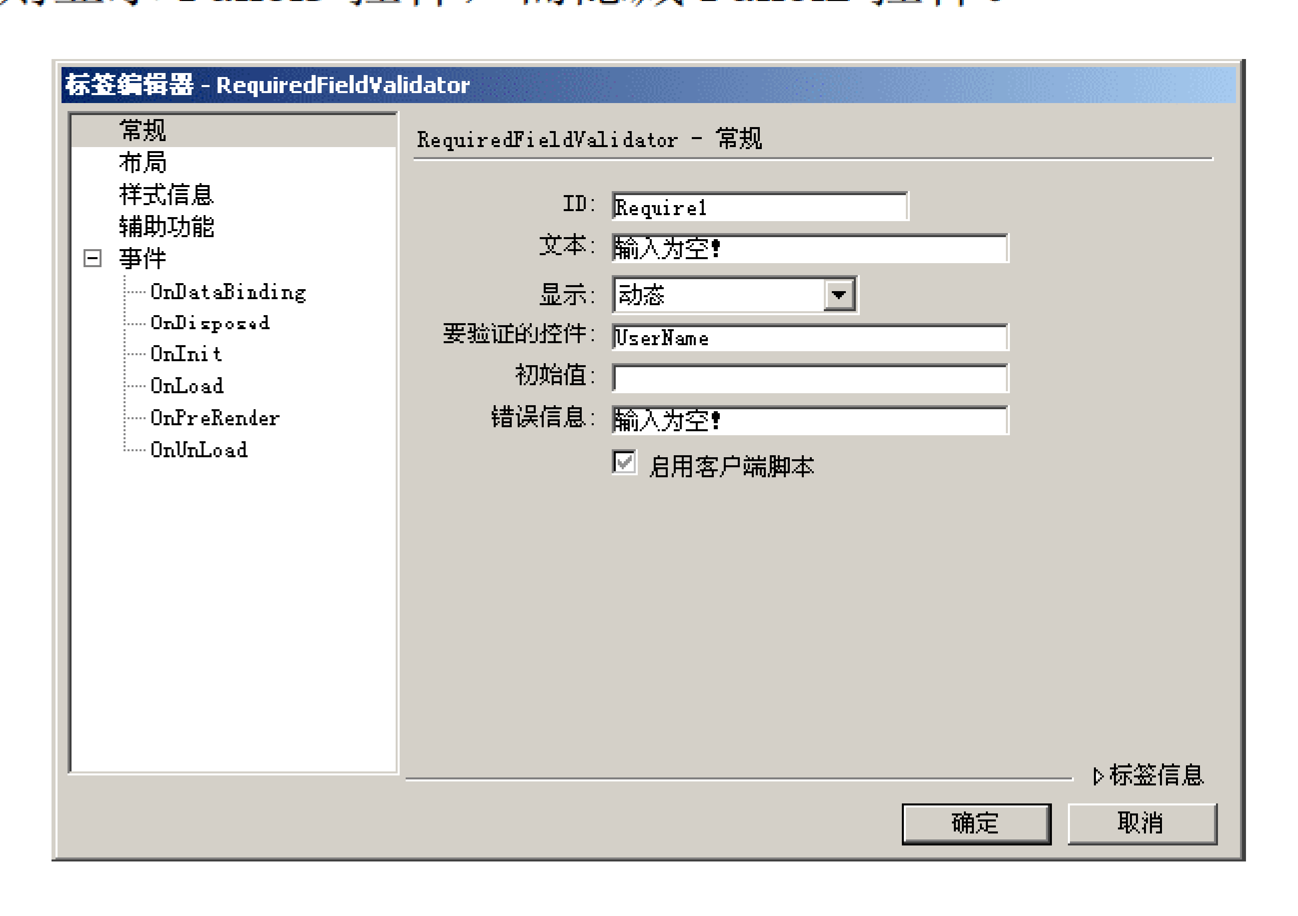


图 8.56　添加控件 Require1

当显示 Panel3 控件时，说明用户经过验证为合法用户，此时将显示用户的密码信息，

提示用户妥善保管好自己的密码，并创建目标为登录页面的链接按钮。

为了判断用户所输入的用户名是否存在，以及用户所输入的出生日期及密码提示问题的答案是否正确，需要创建一个数据集。

打开【应用程序】面板组，切换至【服务器行为】面板，单击+按钮，在弹出的菜单中选择【数据集】命令，在打开的【数据集】对话框中设置数据集的属性，如图 8.57 所示。

图 8.57　创建数据集

数据集的名称设置为 DataSet1，连接选择 Cnn，表格选择 UserSheet，筛选条件设置为字段 LoginName 等于表单变量 UserName。

下面，来看相关事件代码的实现。

【示例代码】

```
<script runat="server">
  '定义共享变量
  Shared theYear As String     '用于存储指定用户的出生日期的年
  Shared theMonth As String    '用于存储指定用户的出生日期的月
  Shared theDay As String      '用于存储指定用户的出生日期的日
  Shared theAnswer As String   '用于存储指定用户的提示问题的答案信息
  '初始化页面加载事件
  Sub Page_Load(ByVal Sender As Object, ByVal E As EventArgs)
    '定义数组 theYear，用于绑定年所对应的下拉列表框控件 Drop_Y
    Dim theYear As New ArrayList
    '定义数组 theMonth，用于绑定月所对应的下拉列表框控件 Drop_M
    Dim theMonth As New ArrayList
    '定义数组 theDay，用于绑定日所对应的下拉列表框控件 Drop_D
    Dim theDay As New ArrayList
    Dim i As Integer
    If (Not Page.IsPostBack) Then
      Panel1.Visible = True   '显示 Panel1
      Panel2.Visible = False  '隐藏 Panel2
      Panel3.Visible = False  '隐藏 Panel3
      '添加年度数据
      For i = 1920 To 2006
        theYear.Add(i.ToString())
Next
'添加月份数据
```

```
        For i = 1 To 12
          theMonth.Add(i.ToString())
  Next
  '添加日数据
        For i = 1 To 31
          theDay.Add(i.ToString())
        Next
        Drop_Y.DataSource = theYear  '绑定下拉列表框 Drop_Y
        Drop_Y.DataBind()
        Drop_M.DataSource = theMonth '绑定下拉列表框 Drop_M
        Drop_M.DataBind()
        Drop_D.DataSource = theDay    '绑定下拉列表框 Drop_D
        Drop_D.DataBind()
      End If
    End Sub
    'Panel1 中【下一步】按钮的单击事件
    Sub Next1_Click(Sender As Object,E As Eventargs)
      '判断数据集筛选结果的记录数是否大于 0
      if DataSet1.DefaultView.Table.Rows.Count>0 then
        '数据集的记录数大于 0，说明用户所输入的用户名称存在，数据集所返回的记录为指定用户
          的相关信息
        '获取数据集中的出生日期所对应的年度
        theYear= Year(DataSet1.DefaultView.Table.Rows(0)("Birth"))
        '获取数据集中的出生日期所对应的月份
        theMonth= Month(DataSet1.DefaultView.Table.Rows(0)("Birth"))
        '获取数据集中的出生日期所对应的日
        theDay= Day(DataSet1.DefaultView.Table.Rows(0)("Birth"))
        '获取数据集中的密码提示问题信息
        Question.Text=DataSet1.DefaultView.Table.Rows(0)("PassQuestion")
        '获取数据集中的用户密码
        Pwd.Text=DataSet1.DefaultView.Table.Rows(0)("UserPass")
        '获取数据集中的密码提示问题答案信息
        theAnswer=DataSet1.DefaultView.Table.Rows(0)("PassAnswer")
        Panel1.Visible = False  '隐藏 Panel1
        Panel2.Visible = True   '显示 Panel2
        Panel3.Visible = False  '隐藏 Panel3
      Else
        '数据集的记录数等于 0，说明用户所输入的用户名称不存在，提示错误信息
        LblErr.Text = "无此用户！"
      End If
    End Sub
    'Panel2 中的【下一步】按钮的单击事件
    Sub Next2_Click(Sender As Object,E As Eventargs)
      '判断用户所选择的年度与出生日期中的年是否一致
      If Trim(theYear) <> Drop_Y.SelectedItem.Text.ToString Then
        LblErr1.Text = "年份错误！"
      '判断用户所选择的月份与出生日期中的月是否一致
      ElseIf Trim(theMonth) <> Drop_M.SelectedItem.Text.ToString Then
        LblErr1.Text = "月份错误！"
      '判断用户所选择的日与出生日期中的日是否一致
      ElseIf Trim(theDay) <> Drop_D.SelectedItem.Text.ToString Then
```

```
        LblErr1.Text = "日期错误！"
      '判断用户所输入的密码提示问题答案是否正确
      ElseIf trim(theAnswer)<>trim(Answer.Text) then
        LblErr1.Text = "提示问题答案错误！"
      Else
        '输入信息正确，显示下一步操作
        Panel1.Visible = False  '隐藏 Panel1
        Panel2.Visible = False  '隐藏 Panel2
        Panel3.Visible = True   '显示 Panel3
      End If
    End Sub
    '【返回登录页面】按钮的单击事件
    Sub Next3_Click(Sender As Object,E As Eventargs)
      '页面跳转至登录页面
      Response.Redirect("default.aspx")
    End Sub
    '定义下拉列表框 Drop_M 的 OnSelectedIndexChanged 事件
    Sub ChangeMonth(ByVal Sender As Object, ByVal E As EventArgs)
      '此事件用于根据用户所选择的年度和月份来确定下拉列表框 Drop_D 中的数据显示
      Dim Days As Integer() = {31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31}
      Dim i As Integer
      Dim theDay As New ArrayList
      If ((Convert.ToInt32(Drop_Y.SelectedItem.Text.ToString()) Mod 4) = 0 And
      (Convert.ToInt32(Drop_Y.SelectedItem.Text.ToString()) Mod 100) <> 0)
      Then
        '当前年度为闰年，将 2 月份所对应的天数改为 29 天
        Days(1) = 29
      End If
      For i = 1 To Days(Convert.ToInt32(Drop_M.SelectedItem.Text.ToString()
      - 1))
        theDay.Add(i.ToString)
      Next
      Drop_D.DataSource = theDay
      Drop_D.DataBind()
    End Sub
  </script>
```

值得一提的是，当单击 Panel2 控件中的【下一步】按钮时，不可再次通过数据集 DataSet1 中的记录来获取指定用户的相关信息。这是因为，数据集 DataSet1 所设置的筛选参数为表单变量 UserName。当单击 Panel1 控件中的【下一步】按钮提交表单数据时，表单变量 UserName 是存在的(即用户名称所对应的文本框的输入值)；而当单击 Panel2 中的【下一步】按钮提交数据时，所提交的表单将不再包括文本框控件 UserName，因此此时将无法通过数据集来获取前面用户所输入的用户名所对应的相关信息。

为了解决这个问题，在代码中定义了 4 个共享变量，分别用于存储后面需要进行比较的用户信息，包括出生日期中的年、月、日以及密码提示问题答案等。当单击 Panel1 控件中的【下一步】按钮时，获取数据集中的相关信息并将其赋值给变量。这样，在单击 Panel2 控件中的【下一步】按钮时，则无须再次从数据集中获取用户信息，而直接将用户的输入与相应的变量值进行比较即可。

页面预览如图 8.58 所示。

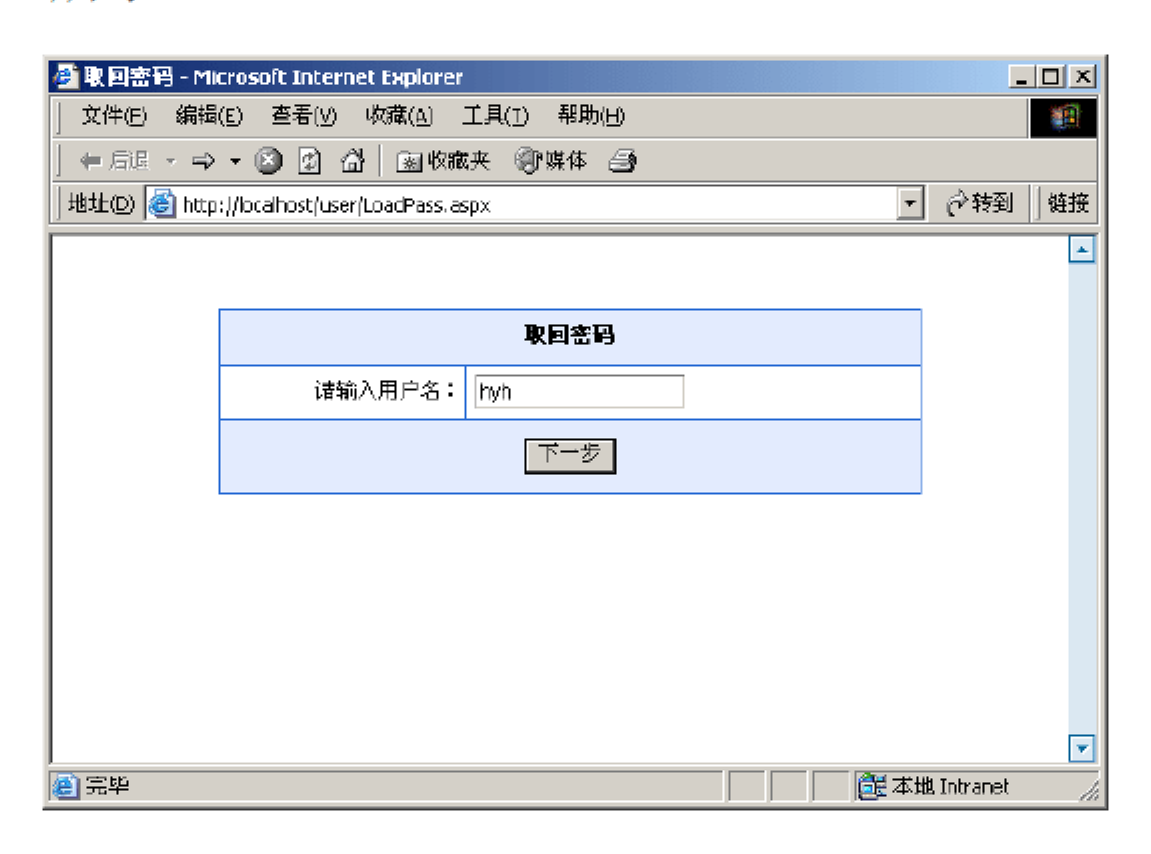


图 8.58　页面预览(1)

输入正确的用户名，单击【下一步】按钮，将显示第二步操作，如图 8.59 所示。

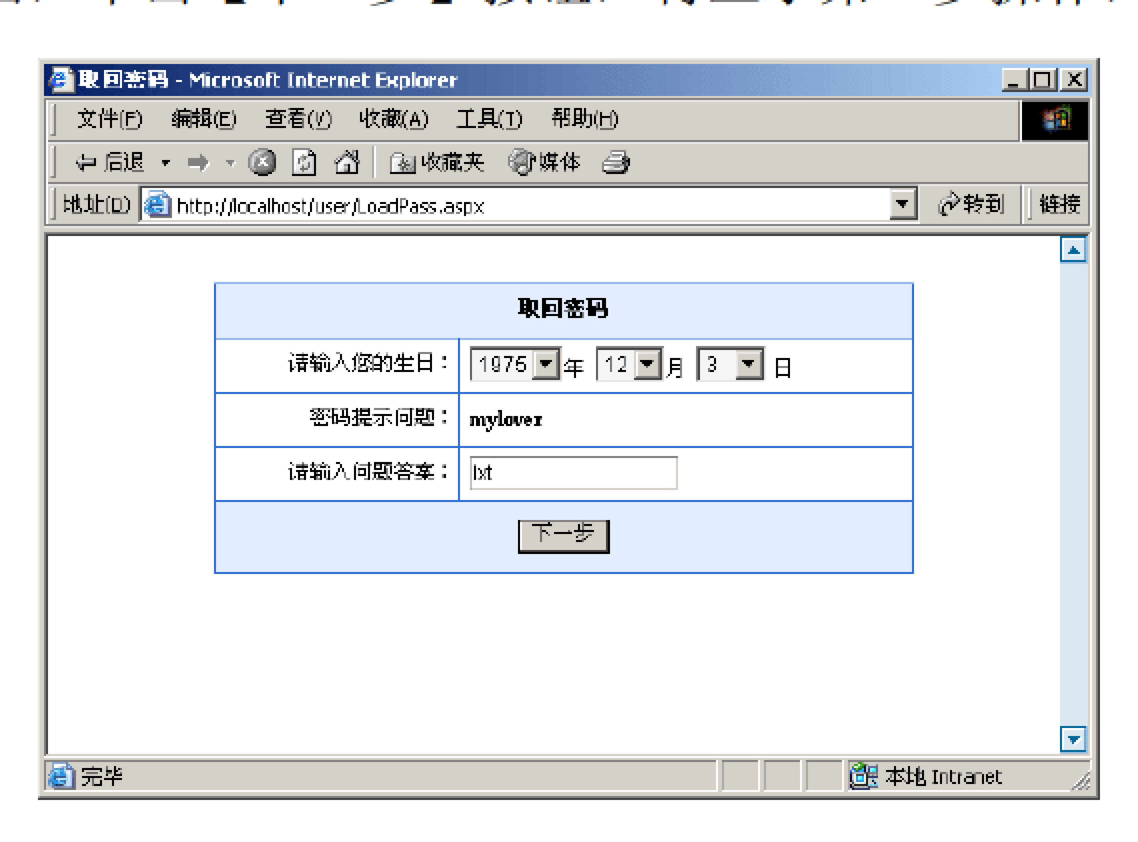


图 8.59　页面预览(2)

选择正确的出生日期，并输入正确的密码提示问题答案，单击【下一步】按钮，此时将会显示该用户的登录密码，如图 8.60 所示。

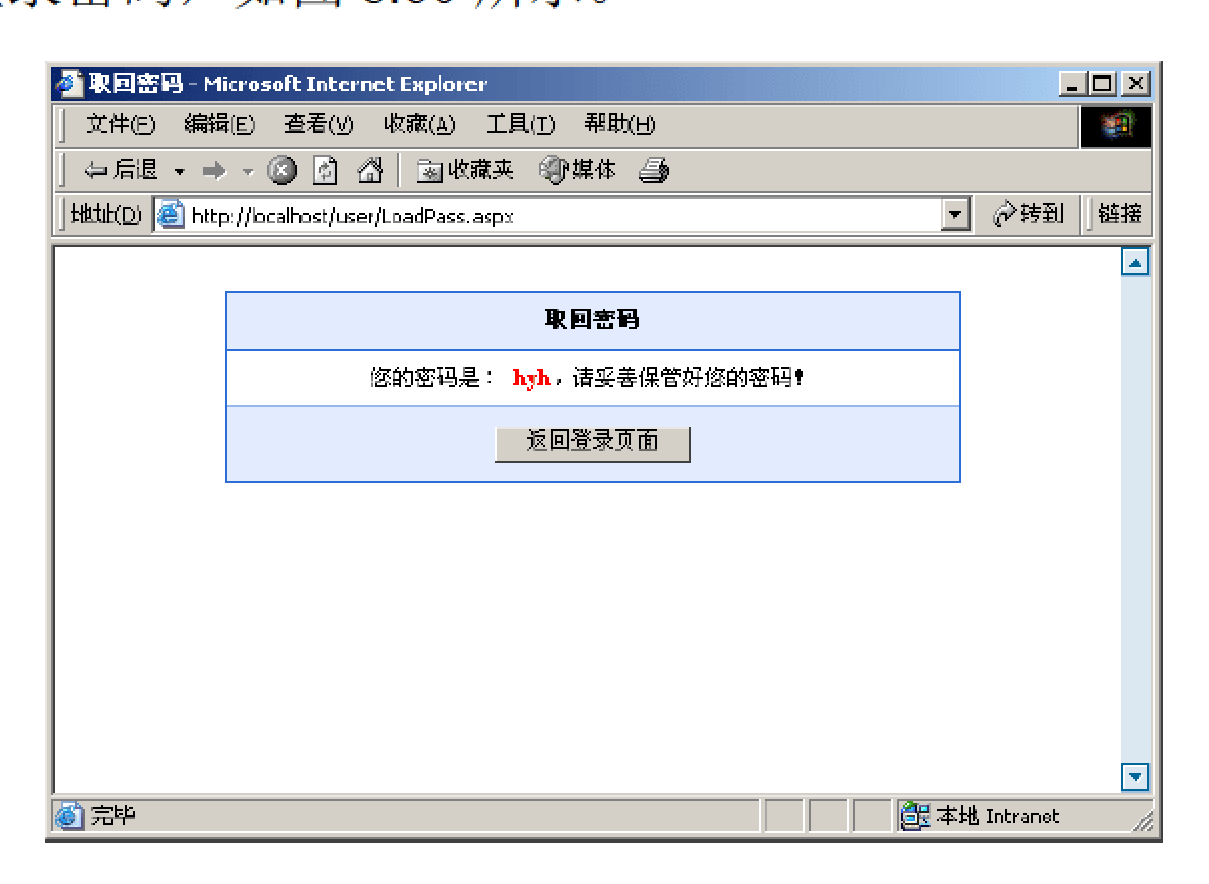


图 8.60　页面预览(3)

至此，找回密码的功能页面设计完成。

8.5　修 改 信 息

当已注册用户的信息发生改变，如电子邮箱发生改变，或想修改密码提示问题的答案等，此时可通过单击主页面上的【修改信息】链接来进入修改信息的页面操作，页面的文件名为 ModiUser.aspx。在修改信息页面中，允许修改除登录名称和登录密码之外的所有信息。

修改信息的页面结构与用户注册中填写个人资料的页面基本类似，不同的是，在修改信息页面中，首先必须获取指定用户的相关信息，并将其填充至相应的文本框控件中，以便用户在原有信息的基础上进行修改操作。

下面，来看页面的表单结构，如图 8.61 所示。

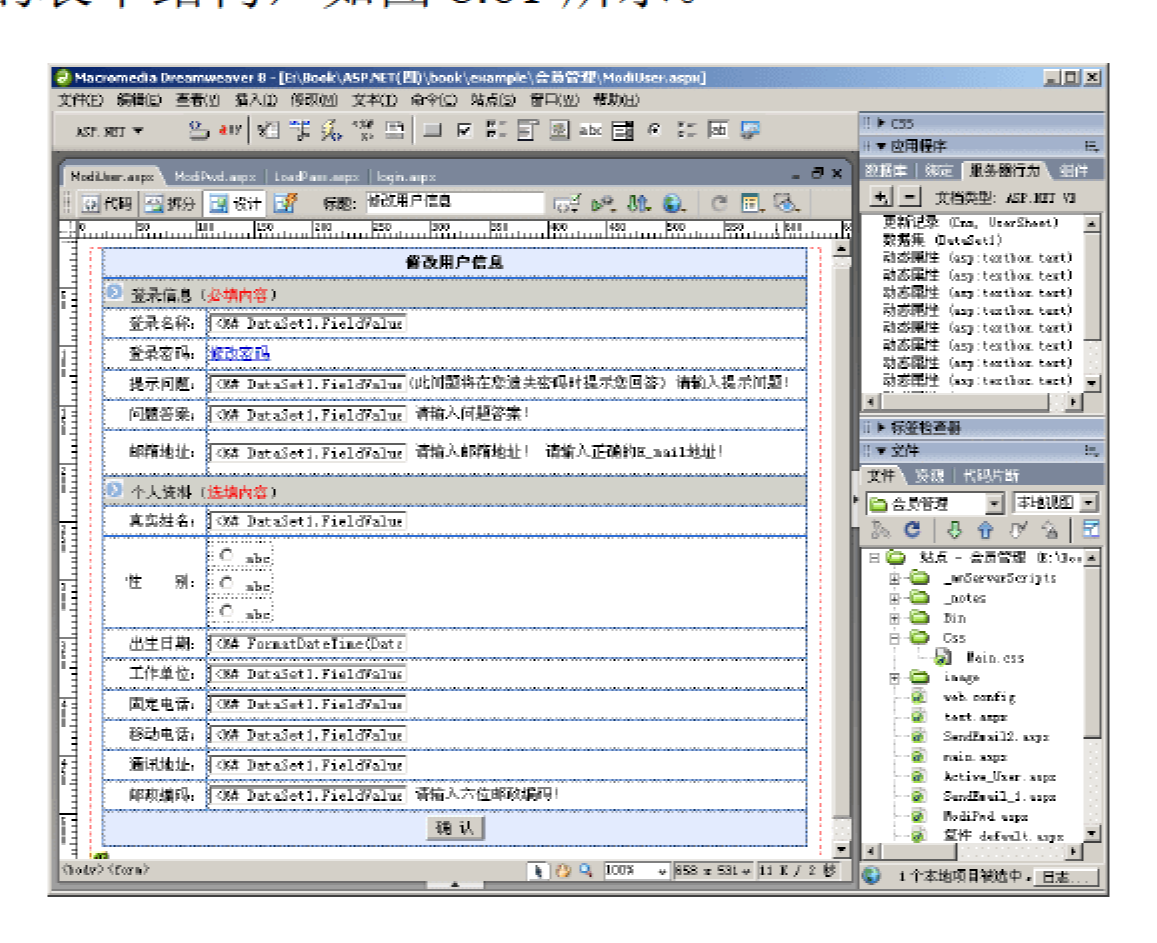


图 8.61　页面设计

在图中，添加了相应的文本和输入框控件以及针对部分文本输入框的验证服务器控件。由于页面的基本结构与填写个人资料的功能页面类似，这里不做过多的说明。为了在页面加载时初始化数据的显示，这里需要创建一个数据集。

相应【数据集】对话框的设置如图 8.62 所示。

图 8.62　创建数据集

在所创建的数据集 DataSet1 中，设置筛选条件为字段 Code 的值等于阶段变量 userid。

这里所谓的阶段变量，就是在登录页面中所创建的 Session 变量。通过此数据集，可获取当前登录用户的相关信息。

在创建了数据集之后，如何在页面加载时将其信息绑定到相应的文本框控件中，这是在创建各个文本框控件所需要注意的问题。下面，以登录名称对应的文本框控件为例来介绍如何在创建文本框控件的同时绑定数据的显示。

(1) 打开【ASP.NET-插入】工具栏，拖动【文本框】控件至相应的位置，此时将弹出【asp:文本框】对话框，如图 8.63 所示。

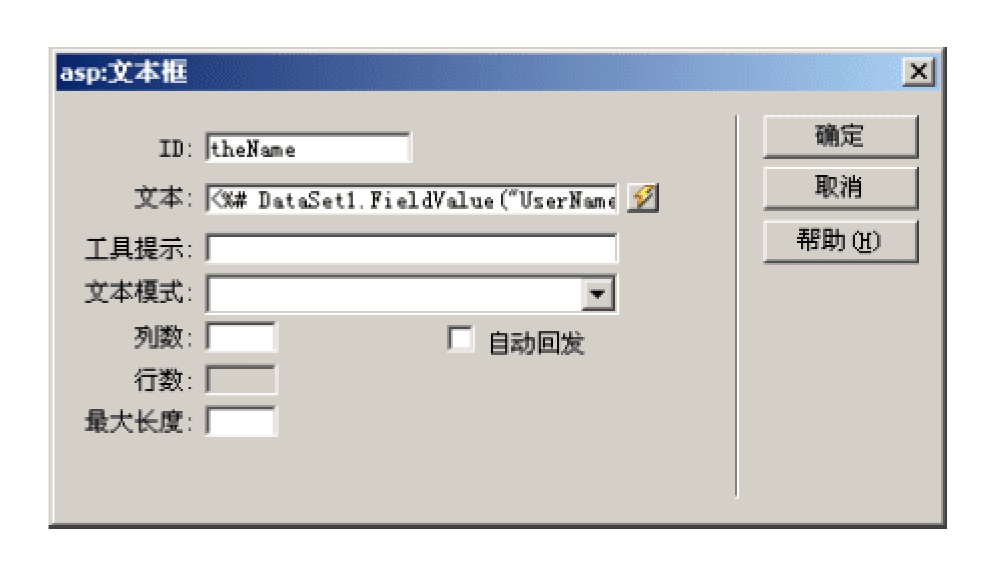


图 8.63　添加文本框控件

(2) 输入 ID 名称为 theName，然后单击按钮，在弹出的【动态数据】对话框中选择数据集 DataSet1，并将其展开，从中选择字段 UserName，如图 8.64 所示。

(3) 单击【确定】按钮，即可完成动态数据的绑定。重复此操作，可添加其他文本框控件，并同时绑定初始化数据的显示。

这里，需要特别提到的是出生日期所对应的文本框控件。由于出生日期在数据表 UserSheet 中所对应的字段 Birth 的数据类型为日期/时间型，如果按以上的操作绑定数据的显示，则在文本框中将会显示完整的日期格式，包括年、月、日和时、分、秒。而对于用户的输入来说，只需要年、月、日即可。此时，我们可在【动态数据】对话框中设置该字段所绑定的数据的显示格式为【日期/时间—2001-1-17】，即仅显示年、月、日，如图 8.65 所示。

在数据表 UserSheet 中，字段 LoginName(登录名称)是表的主键。一般来说，在用户设置登录名称之后，该信息是不允许修改的。为了避免用户在此页面中对登录名称信息进行修改，这里还需要将登录名称所对应的文本框控件 theName 的【只读】属性设置为 True，即不允许用户修改其信息。

图 8.64　绑定动态数据

图 8.65　设置动态数据的显示格式

此外，对于用户的性别信息，这里仍采用 RadioButtonList 控件。而其初始化数据的显示(即默认的选中的项应与当前登录用户的性别一致)将在 Page_Load 事件中通过代码来实现。以下，是 Page_Load 事件的代码：

【示例代码】

```
Sub Page_Load(ByVal Sender As Object, ByVal E As EventArgs)
  If (Not Page.IsPostBack) Then
    '判断 Session 变量 UserID 是否存在
    If IsDBNull(Session("userid")) Or Trim(Session("userid")) = "" Then
      'Session 变量 UserID 不存在，页面跳转至登录页面
      Response.Redirect("default.aspx")
    else
      '判断数据集中当前用户的性别，并初始化 RadioButtonList 控件中默认选中的项
      if DataSet1.DefaultView.Table.Rows(0)("Sex")="男" then
        Sex.Items(0).Selected = True
        Sex.Items(1).Selected = False
      else
        Sex.Items(0).Selected = False
        Sex.Items(1).Selected = True
      end if
    End If
  end if
End Sub
```

在 Page_Load 事件中，还有一项重要的功能，那就是判断 Session 变量 UserID 的值是否存在。当用户正常登录后，Session 变量 UserID 的值是存在的，即登录用户的 ID；而当用户未经登录直接访问此页面时，Session 变量 UserID 是不存在的，此时数据集中的记录为空。由于对用户信息的修改是必须指定当前用户的，如果 Session 变量 UserID 不存在将导致页面无法正常操作。为此，在 Page_Load 事件中首先对 Session 变量 UserID 进行判断，如果该变量不存在则将页面跳转至登录页面，以便用户正常登录后再进行操作。

对所修改的用户信息的具体提交操作如下。

(1)　打开【应用程序】面板组，切换至【服务器行为】面板，单击 + 按钮，在弹出的菜单中选择【更新记录】命令，将弹出【更新记录】对话框，如图 8.66 所示。

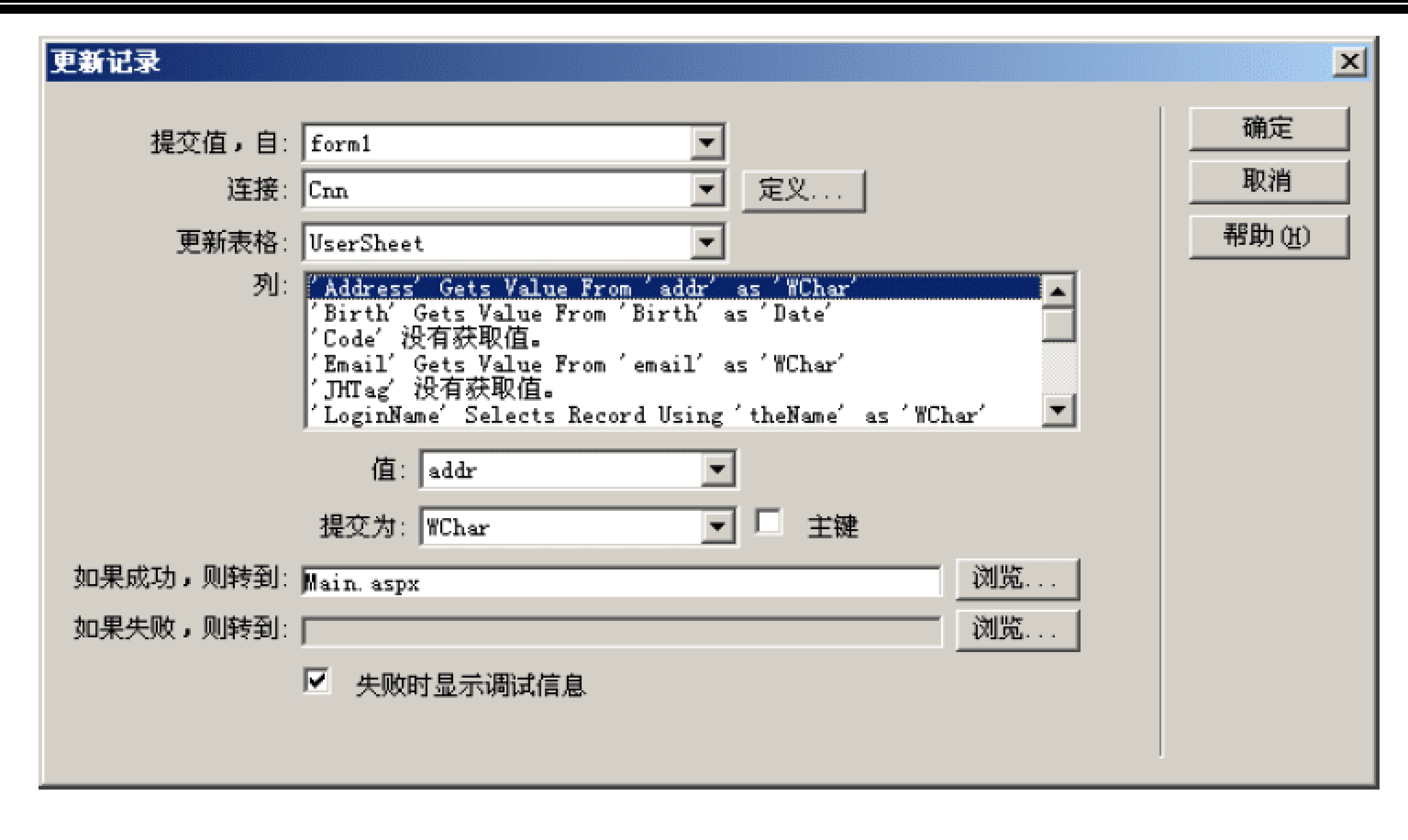


图 8.66　【更新记录】对话框

(2) 在【提交值，自】下拉列表框中选择当前页面所定义的表单元素 form1，数据库连接选择 Cnn，更新表格选择 UserSheet。此时，在【列】列表框中将会列出数据表中的所有字段。这里，对所需更新的字段均需设置其更新的值。当提交表单中的表单元素(如文本框、选择下拉列表框、单选按钮等)存在 ID 名与字段名相同时，系统会自动将其对应起来，即设置该字段的更新值为同名的表单元素的提交值(如图中的 Email 字段与 email 表单元素)。对于不同名的情况，则须进行手工匹配。操作方法是，首先选择字段，然后从【值】下拉列表框中选择作为更新值的表单元素(如图中的 Address 字段与 addr 表单元素)。对于无须进行更新的字段，则可不进行任何操作(如图中的 Code 字段和 JHTag 字段)。

(3) 对于更新成功后，系统自动转向的页面，这里设置为登录后的主页面 Main.aspx。单击【确定】按钮，完成更新记录的设置。

(4) 当用户修改相应的信息后，单击【确认】按钮，即可按照以上的设置执行更新操作。

至此，修改信息页面的功能全部完成，页面预览如图 8.67 所示。

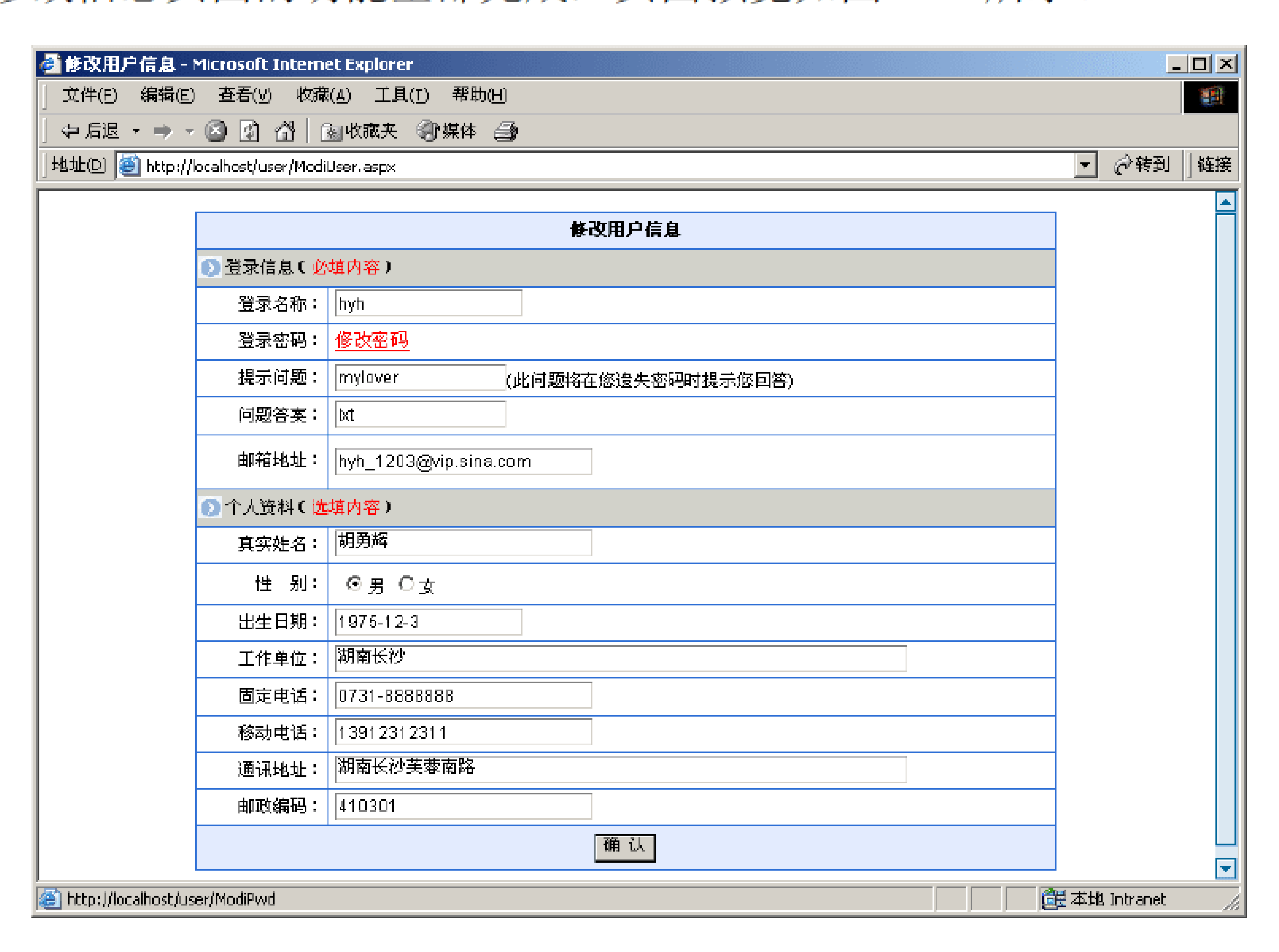


图 8.67　页面预览

8.6　用户密码修改

在修改信息功能中，不包括对用户密码的修改。这是因为在修改用户密码时，用户必须输入原用户密码。只有原密码正确，才能执行更新密码的操作，这也是出于安全方面的考虑。所以，在修改用户信息中，仅提供了对修改密码操作的链接。

对于用户密码的修改，其设计思路是：首先创建数据集，获取当前登录用户的基本信息，包括用户名和原登录密码。然后，创建三个文本框控件，其 ID 分别为 OldPwd、thePass、SurePass，其模式均为密码模式。其中，OldPwd 文本框用于输入当前用户的原密码，thePass 文本框用于输入所要修改的新密码，SurePass 文本框用于再次输入新密码。

对于用户所输入数据的合法性与正确性，均采用验证服务器控件来验证。为了验证用户所输入的旧密码的正确性，需要创建一个与文本框控件 OldPwd 相关联的 CompareValidator 验证服务器控件，其所要比较验证的值为前面所获取的当前用户的原密码。同时，创建一个与文本框控件 thePass 相关联的 RequiredFieldValidator 验证服务器控件，要求在提交数据之前必须输入新密码。为了验证再次输入的新密码与在 thePass 文本框中输入的新密码一致，还需创建一个与文本框控件 PassSure 相关联的 CompareValidator 验证控件，其所要比较验证的控件为文本框控件 thePass。

此外，为了便于更新记录，还需在页面上添加一个隐藏域来存储指定要更新记录的用户名，其取值可从数据集中的字段 LoginName 的值来获取。

(1)　在设计操作界面之前，先添加一个数据集，其属性设置如图 8.68 所示。

图 8.68　添加数据集

其中，名称设为 DataSet1，筛选条件设置为字段 Code 的值等于阶段变量 UserID。

(2)　可设计出修改用户密码的操作界面，如图 8.69 所示。

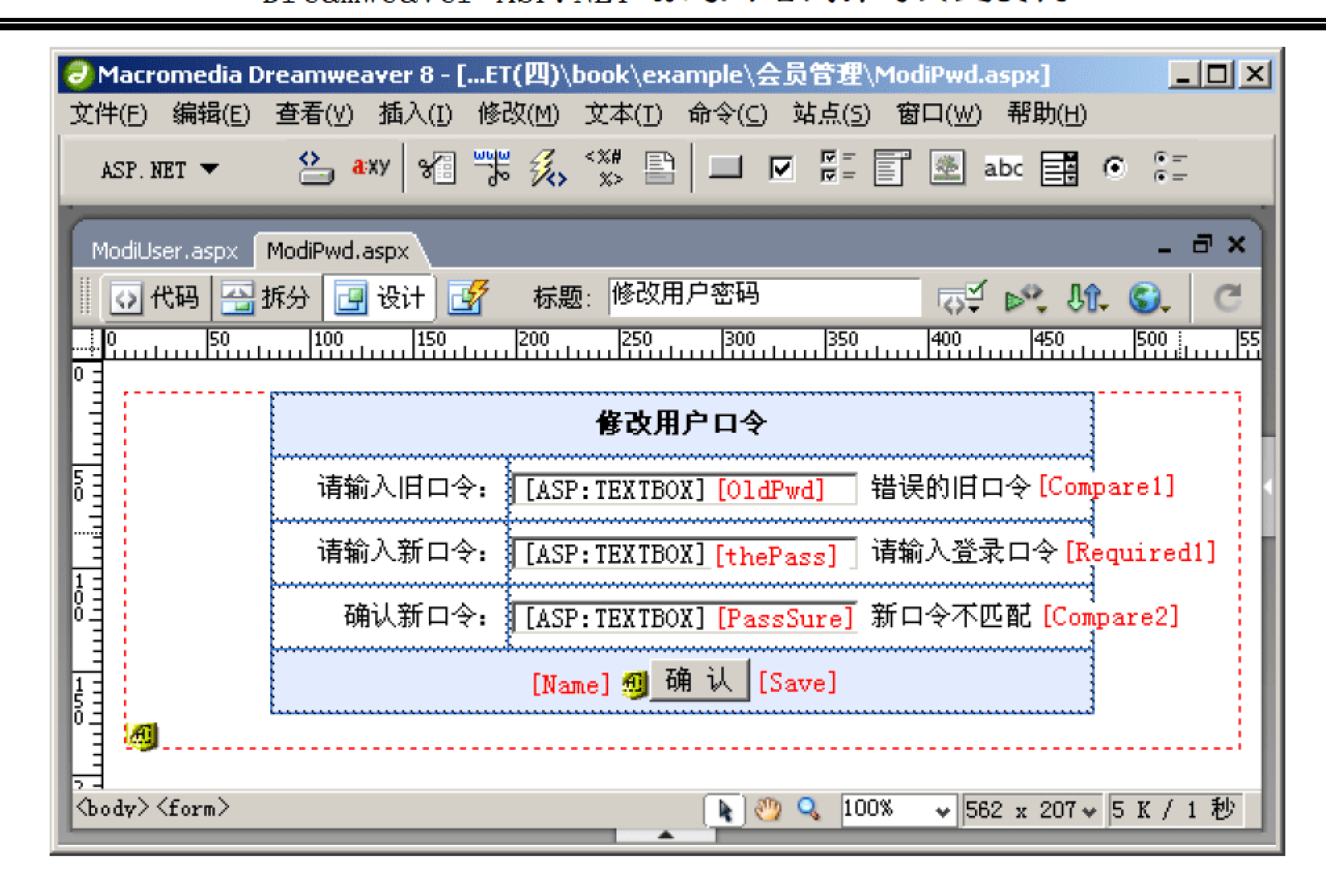


图 8.69　页面设计

在此界面中，验证控件 Compare1 与验证控件 Compare2 虽同为 CompareValidator 控件，但其比较值的类型不同。验证控件 Compare1 是将 OldPwd 文本框中的值与一个常量进行比较，该常量值在 Page_Load 事件中从数据集的 UserPass 字段中获取并指定；而验证控件 Compare2 则是比较两个文本框控件 thePass 和 PassSure 的值是否匹配。

对于隐藏域 Name，主要用于在更新记录时便于识别所要更新的记录，其值设置为<%# DataSet1.FieldValue("LoginName", Container) %>，即数据集中 LoginName 字段的值。

对于数据的更新，仍采用 Dreamweaver 8 中【服务器行为】面板的【更新记录】命令来实现。

打开【应用程序】面板组，切换至【服务器行为】面板，单击+按钮，在弹出的菜单中选择【更新记录】命令，将弹出【更新记录】对话框，如图 8.70 所示。

这里，所需设置对应值的列仅包括 LoginName 和 UserPass。其中，字段 UserPass 是所要更新的数据，其取值设为文本框控件 thePass 的值；字段 LoginName 是用于识别所要更新的记录的主键，其取值设为隐藏域 Name 的值。单击【确定】按钮，完成设置。

切换至【代码】视图，这里还需添加页面的 Page_Load 事件，其代码如下：

【示例代码】

```
Sub Page_Load(ByVal Sender As Object, ByVal E As EventArgs)
  If (Not Page.IsPostBack) Then
    '判断 Session 变量 UserID 是否存在
    If IsDBNull(Session("userid")) Or Trim(Session("userid")) = "" Then
      'Session 变量 UserID 不存在，页面跳转至登录页面
      Response.Redirect("default.aspx")
    Else
      If DataSet1.DefaultView.Table.Rows.Count>0 Then
       '当数据集的记录不为空时，设置 Compare1 控件的比较值为数据集中的 UserPass 字段值
       Compare1.ValueToCompare =
```

```
DataSet1.DefaultView.Table.Rows(0)("UserPass")
      End if
    End If
  End If
End Sub
```

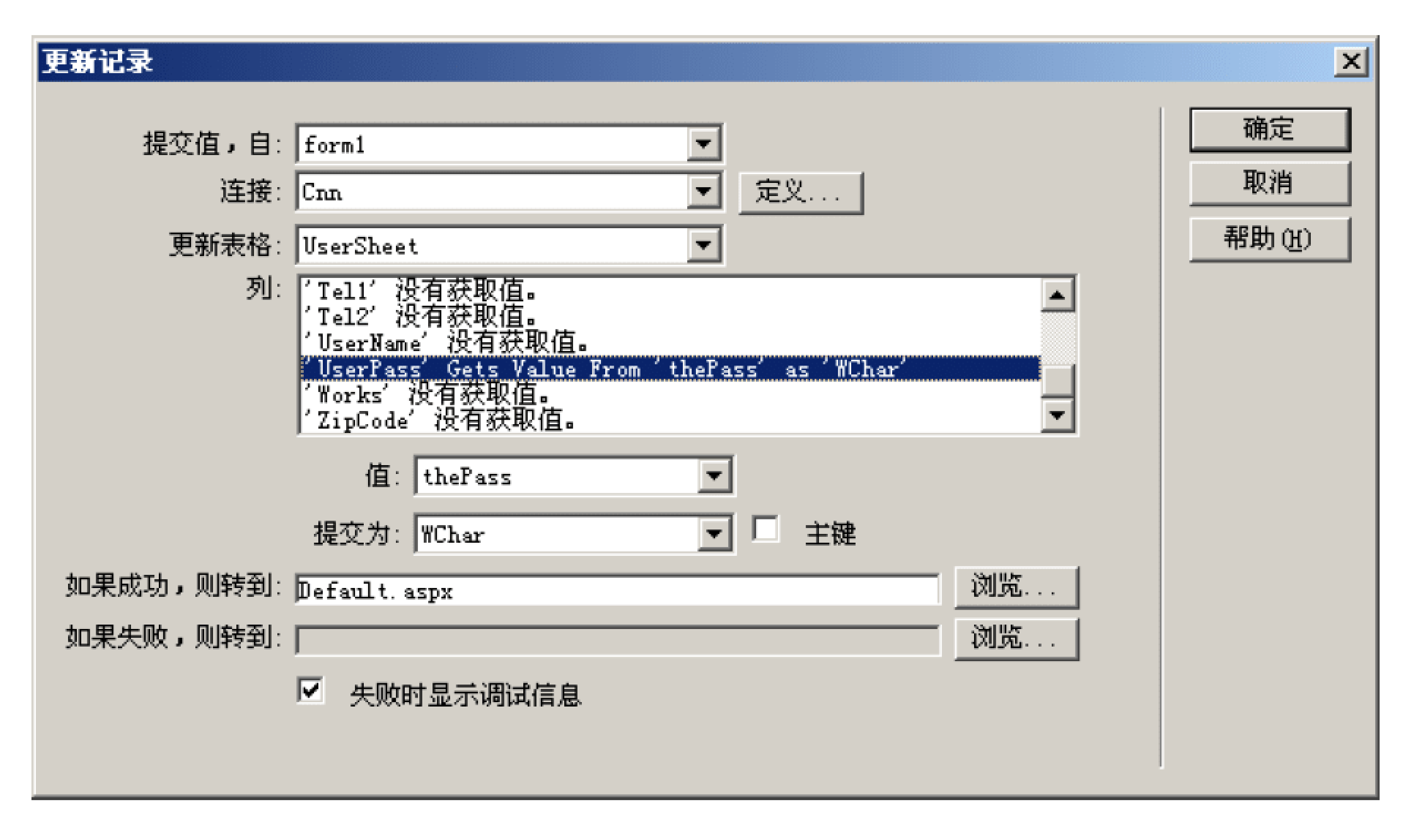


图 8.70　【更新记录】对话框

至此，修改用户密码的页面功能全部完成，页面预览效果如图 8.71 所示。

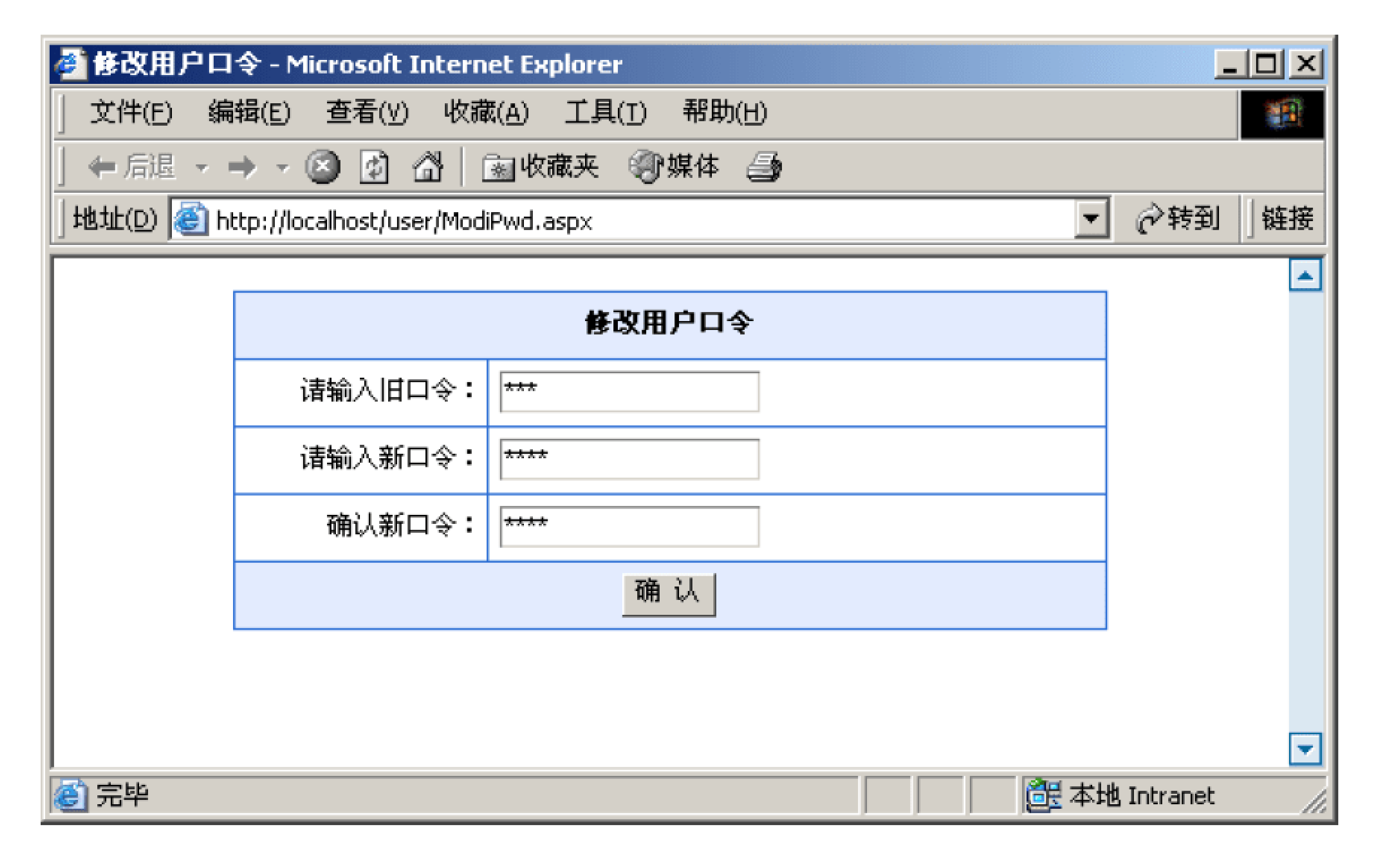


图 8.71　页面预览效果

8.7　习　　题

(1)　练习 ASP.NET 中的验证服务器控件的使用。

(2)　用户注册页面中，在发送电子邮件之前添加对是否连接 Internet 的判断或添加错误

控件，避免因发送电子邮件失败而导致页面异常。

(3) 修改找回密码功能，当用户输入的出生日期和密码提示问题答案均正确时，直接将密码发送至用户所注册的邮箱中，并提示用户接收邮件。

第 9 章　网 络 相 册

网络相册是为个人用户提供的具有个性化的展示平台，它可以提供对个人相片的网络存储、分类管理和对外展示。伴随着网络的发展，网络相册也逐渐成为网上冲浪者的新宠。

9.1　系 统 分 析

9.1.1　系统功能

对于网络相册来说，对外展示、提供浏览是其最基本的功能。浏览者可直接查看或分类查看用户上传的所有相片。同时，对于感兴趣的相片，还可查看该相片的他人留言或发表自己的留言信息。

通过网络相册的后台，用户可自定义相片分类信息，上传自己的相片，同时可对已上传的相片进行管理，包括对相片信息的修改、删除和对相片留言的管理。

作为网络相册的辅助功能，需要对相片的点击和留言进行统计，统计出来的点击数和留言数在一定程度上说明了浏览者对照片的关注和喜爱程度。

在本例中，将网络相册的功能分为前台页面功能和后台页面功能。

1. 前台页面功能

前台页面主要针对一般浏览者，其功能包括发布浏览相册、查看指定相片及相关留言、发表留言等。

1)　浏览相册

在网络相册的主页面，可通过选择相片的分类来浏览指定分类下的所有相片。浏览相片的同时，可查看相片的点击数和留言数。

2)　查看指定相片

对于感兴趣的相片，可通过点击来放大该相片的显示，同时还可查看与该相片相关的留言。

3)　发表留言

对于感兴趣的相片，可发表自己的留言。

2. 后台页面功能

后台页面主要针对管理员，其功能包括添加分类、管理分类、上传相片、管理相片和管理相片留言等。

1)　添加分类

允许添加相片的分类信息。

2)　管理分类

对相片的分类信息进行管理，包括对指定分类的编辑和删除。当指定的分类下存在相片信息时，不可执行对该分类的删除操作。

3)　上传相片

允许上传指定的相片。在上传相片的同时，需指定相片名称和相片的所属分类。同时，提供对相片的预览功能。

4)　管理相片

对所上传的相片进行管理，包括对相片信息的修改和删除以及对指定相片的留言进行管理的操作链接。同时，提供对相片的预览功能。

5)　管理相片留言

对指定相片的留言信息进行管理，包括对所有留言的浏览和对指定留言的删除操作。

9.1.2　数据库的建立

本系统使用 Microsoft Access 2000 类数据库，其数据库名为 Data.Mdb，在该数据库中添加了 ClassInfo(分类信息表)、PhotoInfo(相片信息表)、PhotoLY(相片留言信息表)和 UserInfo(管理员信息表) 4 个数据表。

1. 分类信息表(ClassInfo)

ClassInfo 数据表主要用于存储相片的分类信息，其表结构见表 9.1。

表 9.1　ClassInfo 数据表

字 段 名	数据类型	字段描述	字段大小	默 认 值	主　键
ID	自动编号	分类编号	长整型		P
ClassName	文本	分类名称	50		

2. 相片信息表(PhotoInfo)

PhotoInfo 数据表主要用于存储相片的基本信息，其表结构见表 9.2。

表 9.2　PhotoInfo 数据表

字 段 名	数据类型	字段描述	字段大小	默 认 值	主　键
ID	自动编号	相片编号	长整型		P
PicAddr	文本	相片地址	100		
PicName	文本	相片名称	100		
ClassID	数字	所属相片分类的编号			
DDate	日期/时间	相片的上传时间		Now()	
ClickNum	数字	点击次数		0	
Descr	备注	相片描述			

说明：字段 Ddate 的默认值为系统函数 Now()，该函数用于获取系统的当前时间。字段 ClassID 表示所属分类编号，该字段与 ClassInfo 数据表中的字段 ID 相关联。

3. 相片留言信息表(PhotoLY)

PhotoLY 数据表主要用于存储相片的留言信息，其表结构见表 9.3。

表 9.3　PhotoLY 数据表

字 段 名	数据类型	字段描述	字段大小	默 认 值	主　键
ID	自动编号	留言编号	长整型		P
PhotoID	数字	相片编号			
LYName	文本	留言人名称	50		
LYContent	备注	留言内容			
LYTime	日期/时间	留言时间		Now()	

说明：字段 PhotoID 表示留言所对应的相片编号，该字段与 PhotoInfo 数据表中的字段 ID 相关联。

4. 管理员信息表(UserInfo)

UserInfo 数据表主要用于存储管理员的基本信息，其表结构见表 9.4。

表 9.4　UserInfo 数据表

字 段 名	数据类型	字段描述	字段大小	默 认 值	主　键
ID	自动编号	管理员编号	长整型		P
UserName	文本	管理员名称	50		
Pwd	文本	管理员密码	50		

9.1.3　站点设置

本例中，站点设置的本地信息如图 9.1 所示。

其中，站点名称设置为“网络相册”，本地根文件夹设置为网络相册系统所在的磁盘目录，默认图像文件夹则设置为网络相册系统下的 Image 目录，HTTP 地址设置为“http://localhost/wlxc”(在此之前，需将网络相册系统设置为 Web 共享，并设置其共享名为 wlxc)。

站点设置中，远程信息设置如图 9.2 所示。

由于对网络相册系统的设计、测试和运行均是在本机上操作的，因此这里将访问方式设置为【本地/网络】，而远端文件夹则与本地根文件夹设置为相同的目录。

站点设置中，测试服务器设置如图 9.3 所示。

对于站点的部署，请参考本书第 7 章，这里不再赘述。

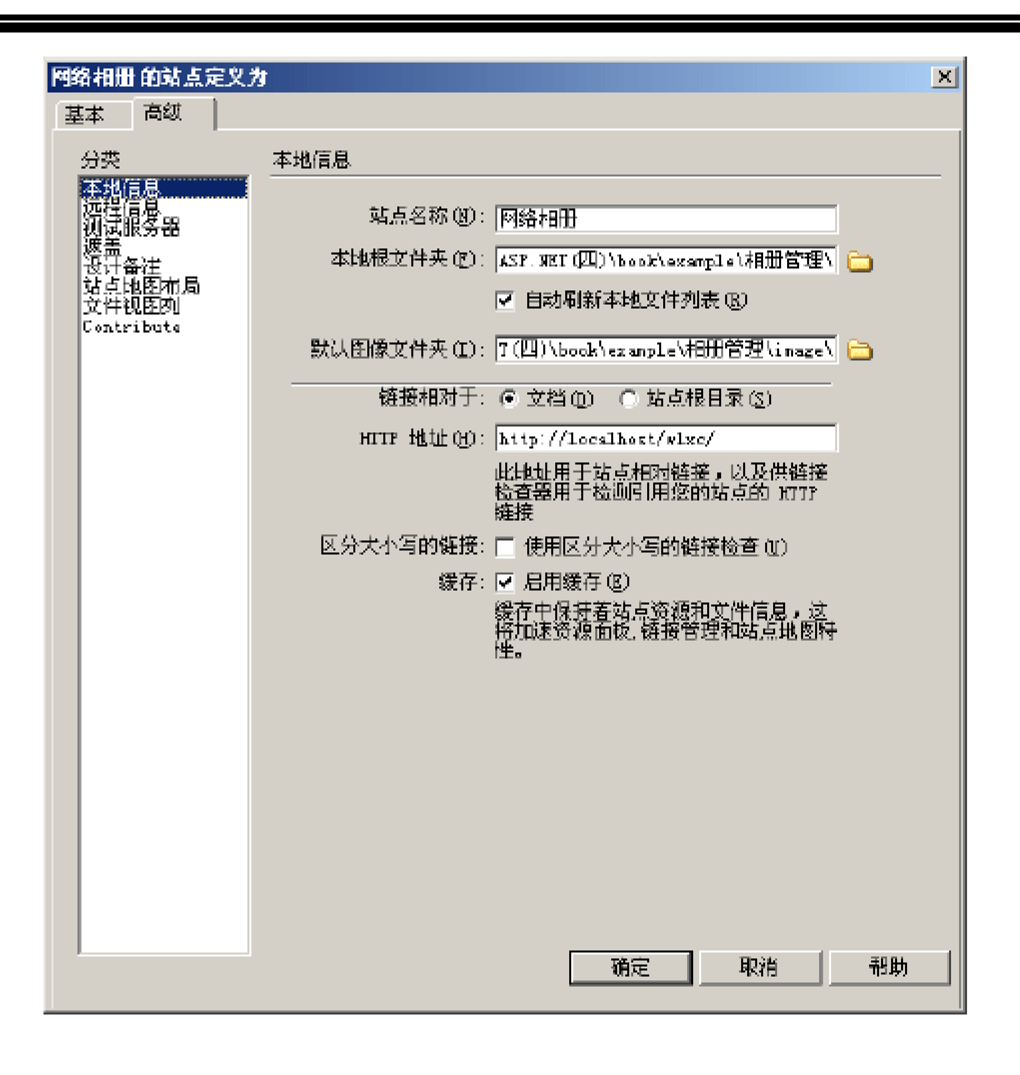


图 9.1　本地信息

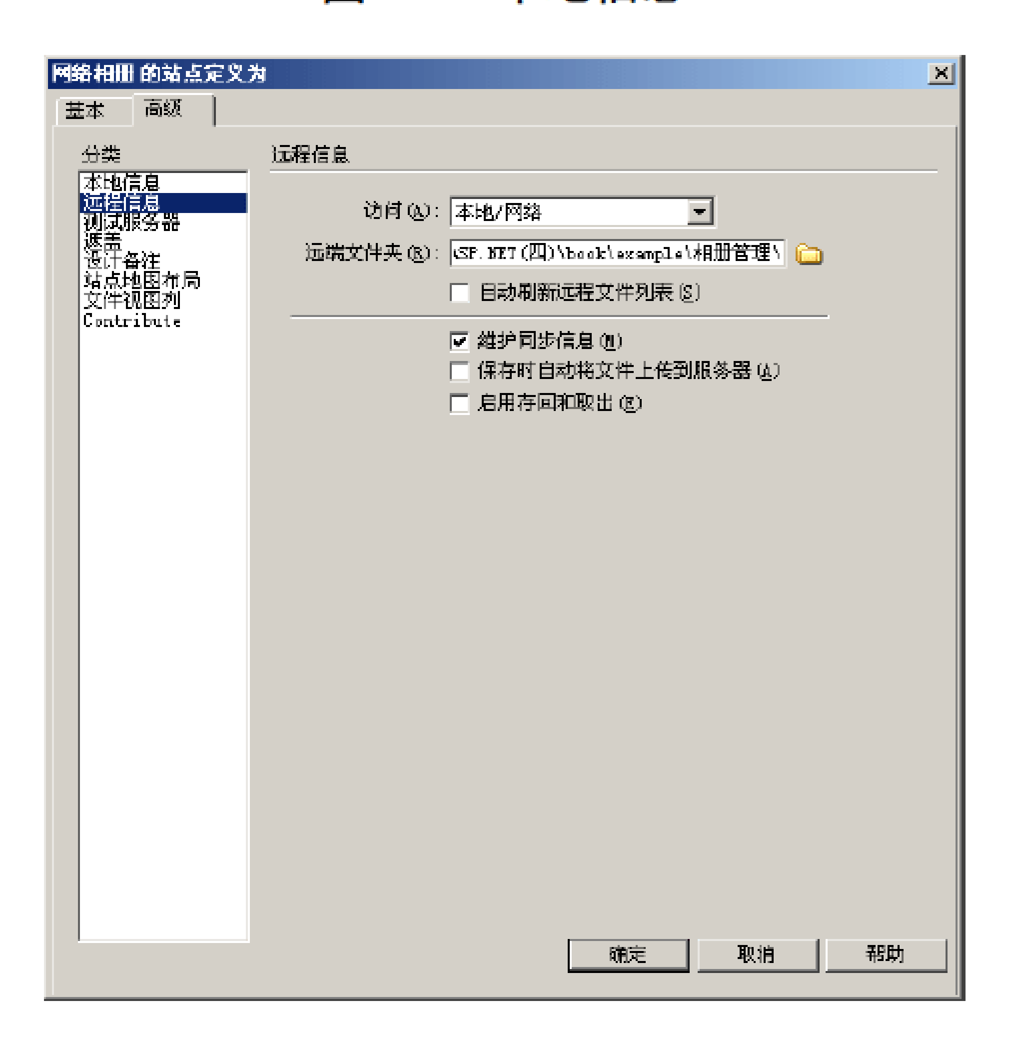


图 9.2　远程信息

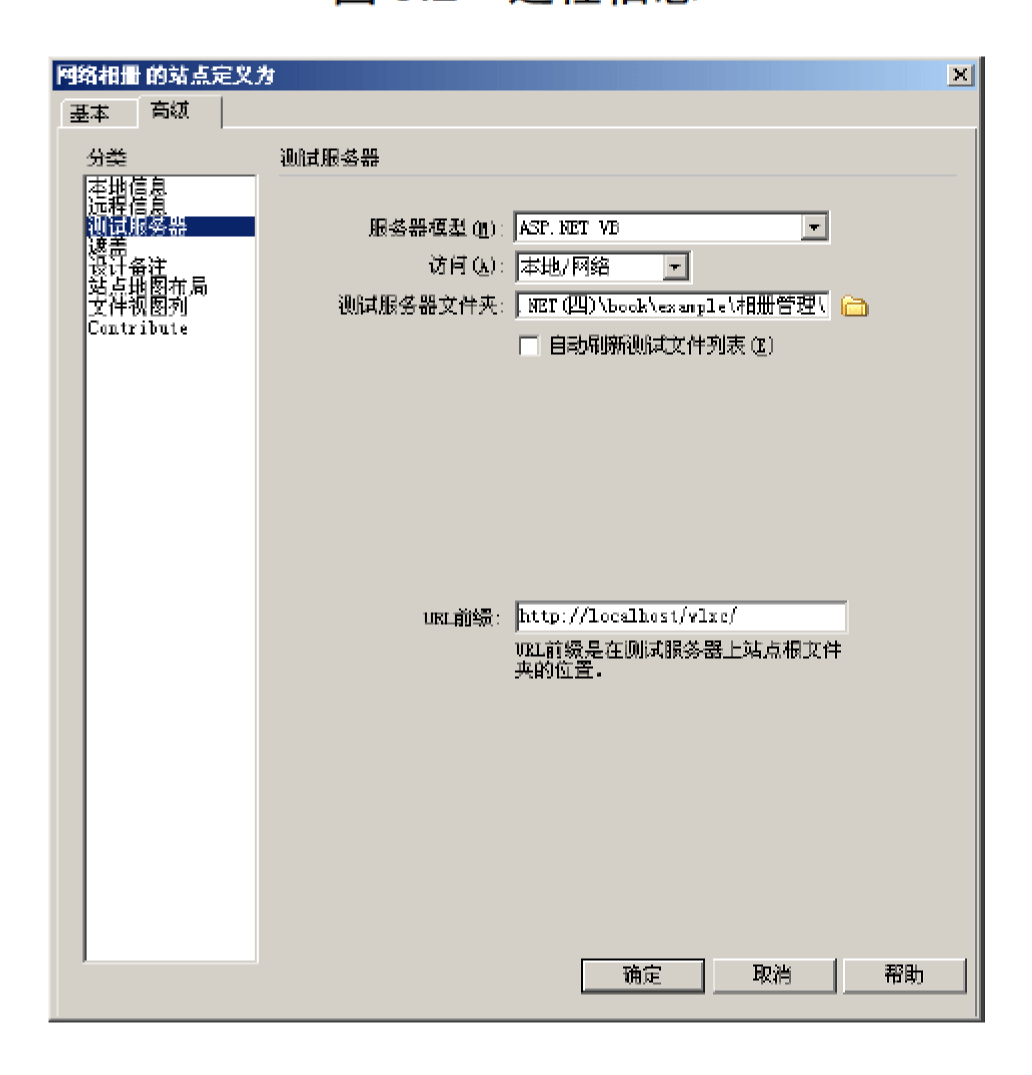


图 9.3　测试服务器

9.2　前台功能实现

网络相册的前台功能包括浏览相册、查看相片和发表留言。其中，浏览相册页面所对应的文件为 Default.aspx，查看相片和发表留言操作被安排在同一页面，其文件名为 ViewPhoto.aspx。

9.2.1　构建模板页

在实现前台页面的功能之前，先来为前台的页面构建一个模板。模板的用途是构建一个基本的页面框架布局，从而使得其他页面可在此基础上快捷生成，并使整体的布局保持一致。此外，通过更新模板，可以更新建立在此模板上的所有页面，这也是模板最强大的用途之一。

(1) 选择【文件】|【新建】命令，在弹出的【新建文档】对话框中选择【模板页】|【ASP.NET VB 模板】选项，如图 9.4 所示。单击【创建】按钮，即可新建一个模板页。

图 9.4　【新建文档】对话框

(2) 导入外部样式表。打开 CSS 面板并右击，在弹出的快捷菜单中选择【附加样式表】命令，此时将弹出【链接外部样式表】对话框，如图 9.5 所示。

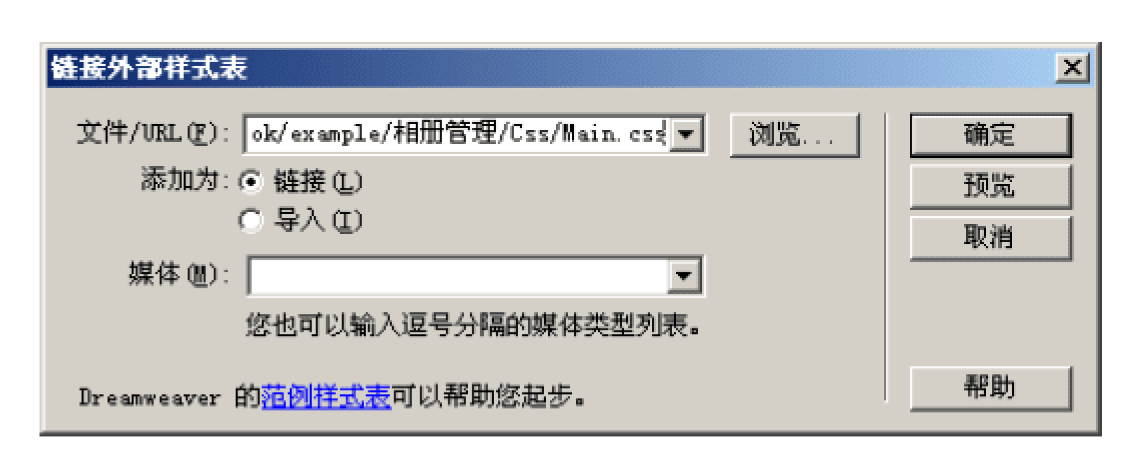


图 9.5　【链接外部样式表】对话框

(3) 单击【浏览】按钮，选择系统目录中的 CSS 文件夹下的 Main.css 文件，设置添加方式为【链接】。单击【确定】按钮，完成样式表的附加。

(4) 切换至【设计】视图，单击【属性】面板中【页面属性】按钮，在弹出的【页面

属性】对话框中，设置页面的背景颜色值为“#d2d2ca”，上边距为 0，如图 9.6 所示。

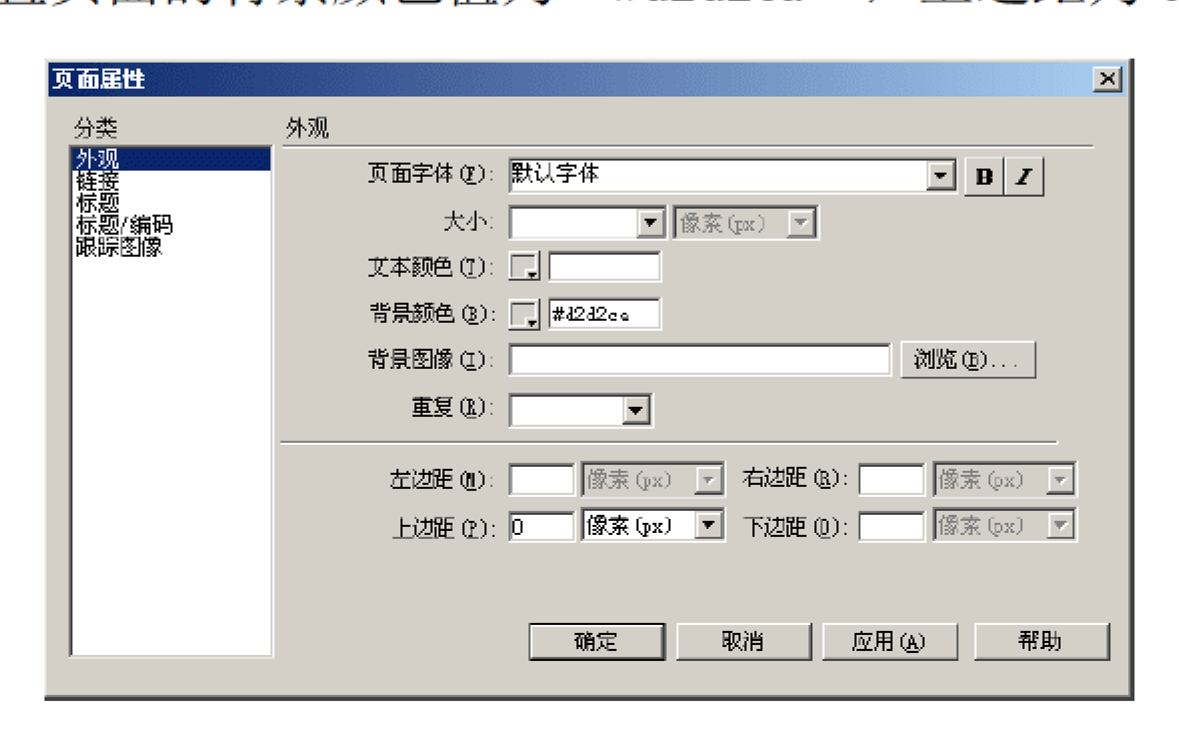


图 9.6　【页面属性】对话框

(5) 单击【确定】按钮，返回设计窗口。选择【插入】|【表格】命令，在页面上插入一个 3 行 1 列的表格 table1，其属性设置如图 9.7 所示。

图 9.7　表格属性的设置

(6) 在表格 table1 的第 1 行中，插入一个图像，其源文件设置为 Image 文件夹下的 Top.gif。在第 2 行中，设置其背景颜色值为“#F8FAFC”，同时创建一个链接，其文本为“后台管理”，链接地址为“../Admin.aspx”。在表格 table1 的第 3 行中，插入一个 1 行 2 列的嵌套表格 table2，其属性设置如图 9.8 所示。

图 9.8　嵌套表格属性的设置

(7) 在表格 table2 中，设置第 1 列的单元格宽度为 160，第 2 列的单元格宽度为 620，同时设置两个单元格的背景颜色值为“#FFFFFF”。

将光标置于表格 table2 的第 1 个单元格中，选择【插入】|【模板对象】|【可编辑区域】命令，此时将弹出【新建可编辑区域】对话框，如图 9.9 所示。

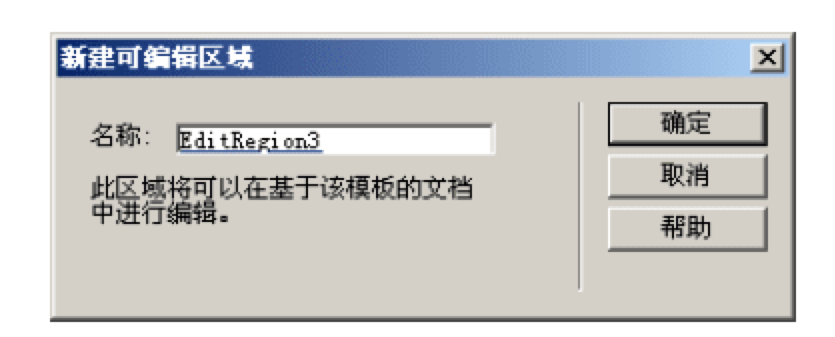


图 9.9　【新建可编辑区域】对话框

设置名称为 EditRegion3，单击【确定】按钮，完成可编辑区域的添加。在表格 table2 的第 2 个单元格中，重复以上操作，添加名称为 EditRegion4 的可编辑区域。

(8) 选择【文件】|【保存】命令，将页面命名为“前台模板”，并加以保存。此时，系统会自动在网络相册的系统目录下新建一个名为 Templates 的文件夹，并在其中生成一个名为“前台模板.dwt.aspx”的页面文件，这就是刚才创建的模板。

至此，前台功能页面的模板创建完成，如图 9.10 所示。

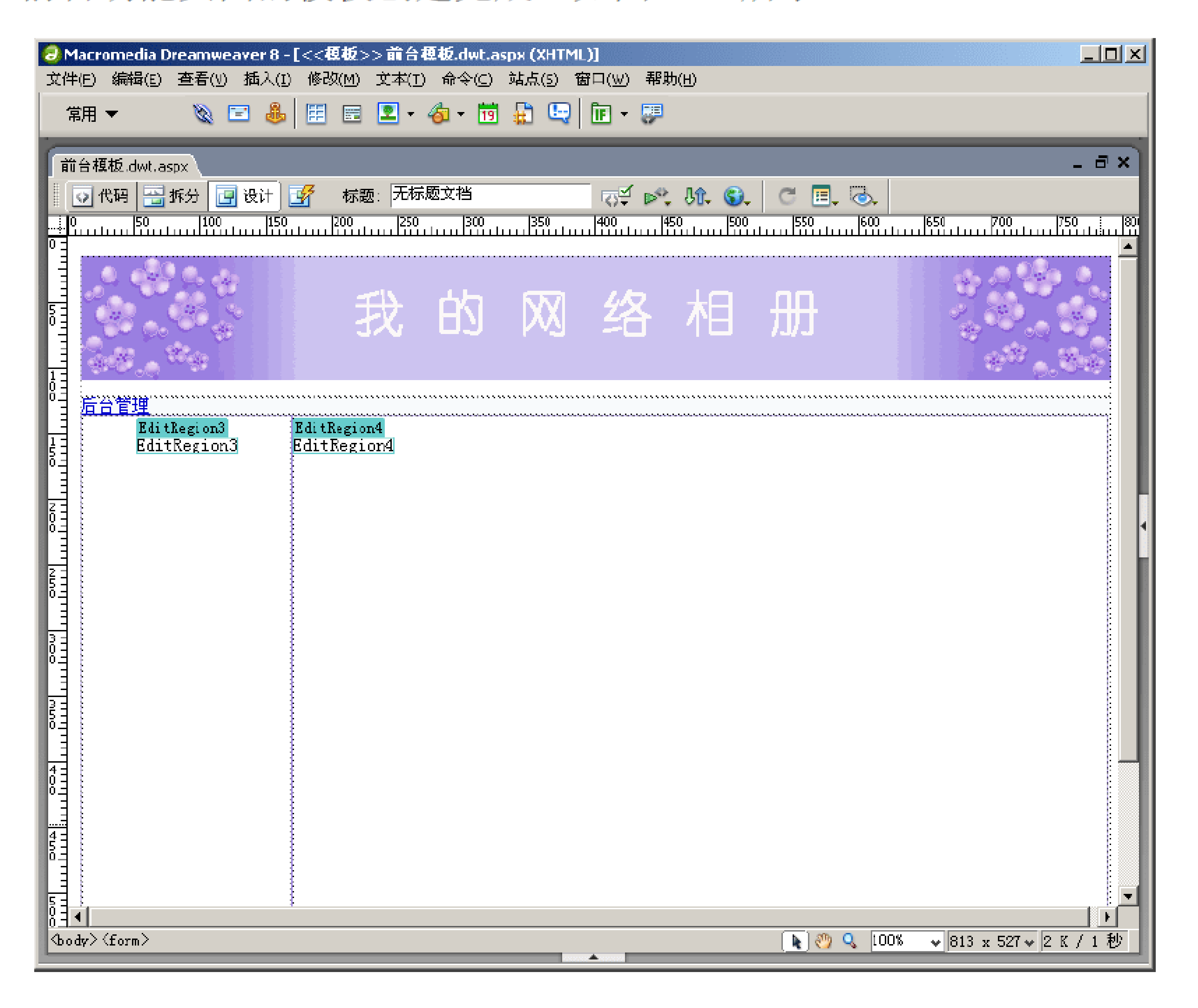


图 9.10　前台功能页面模板

9.2.2　浏览相册

浏览相册是用户进入网络相册系统的首页面。在该页面中，首先必须罗列出所有的相片分类，以供用户单击选择；然后需要显示某一分类下的所有相片以供浏览。在相片的浏览中，除了显示相片外，还需显示相片的附加信息，包括相片名称、单击次数、留言次数和相片描述等。

1. 页面基本结构的实现

下面，来看一看该页面的具体实现。

(1) 新建一个 ASP.NET VB 类型的页面(动态页)，将其命名为 Default.aspx。选择【修改】|【模板】|【套用模板到页】命令，在弹出的【选择模板】对话框中，选择前面所创建的【前台模板】，如图 9.11 所示。并单击【选定】按钮，将模板应用到本页面。

(2) 将光标移至 EditRegion3 可编辑区域中，添加文本“相片分类”，设置其字体大小为 14 像素，并加粗显示。再将光标移至 EditRegion4 可编辑区域中，添加文本“当前分类：”，设置其字体为 14 像素，加粗显示，并在该文本下添加一条水平线，即<hr>标签。

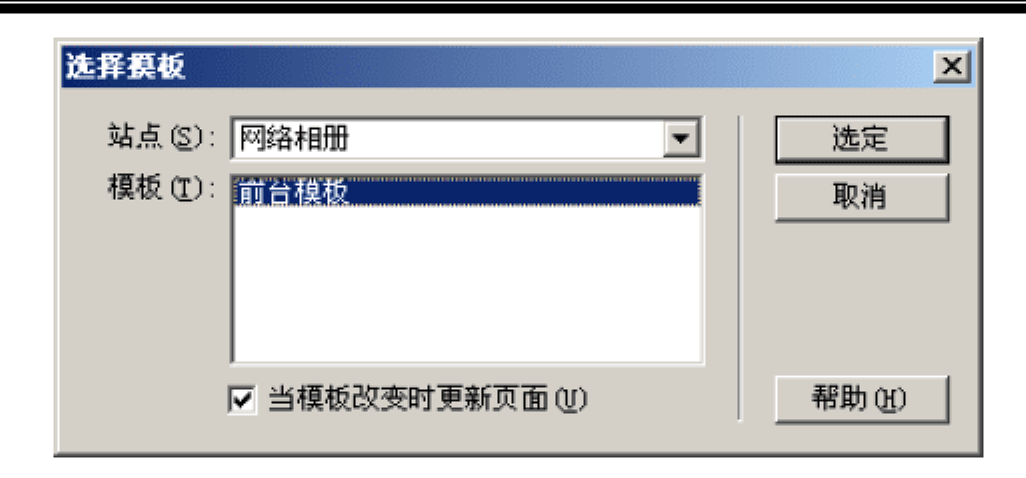


图 9.11　选择模板

页面基本结构如图 9.12 所示。

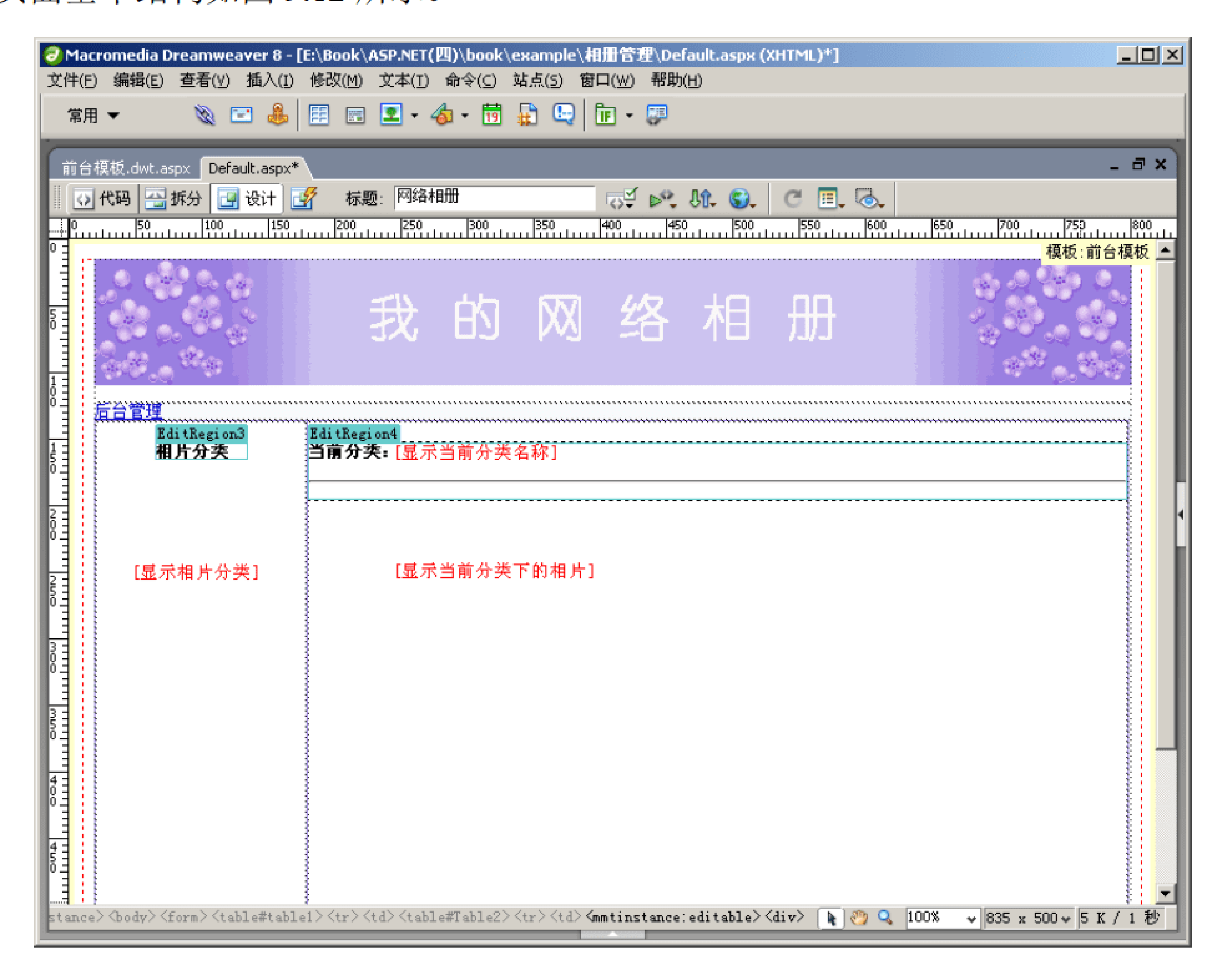


图 9.12　页面基本结构

2. 建立数据库连接

对于相片分类的显示可通过 Dreamweaver 8 中提供的【服务器行为】面板中的【重复区域】命令来实现，即采用 ASP.NET 中的 Repeater 数据控件。而对于指定分类下的相片浏览，则可通过【服务器行为】面板中的【数据列表】命令来实现，即 ASP.NET 中的 DataList 数据控件。为此，需要分别创建两个数据集。

在创建数据集之前，先来建立数据库连接。

(1) 打开【应用程序】面板组，切换至【数据库】面板，单击 按钮，从弹出的菜单中选择【OLE DB 连接】命令，在弹出的【OLE DB 连接】对话框中设置连接名称为 Cnn，如图 9.13 所示。

图 9.13　创建数据库连接

(2) 单击【建立】按钮，在 Provider 选项卡中选择 Microsoft Jet 4.0 OLE DB Provider。然后切换至 Connection 选项卡，单击 ... 按钮，选择网络相册系统目录下的 Data 子目录下的 Data.mdb 文件作为系统数据库的文件，如图 9.14 所示。

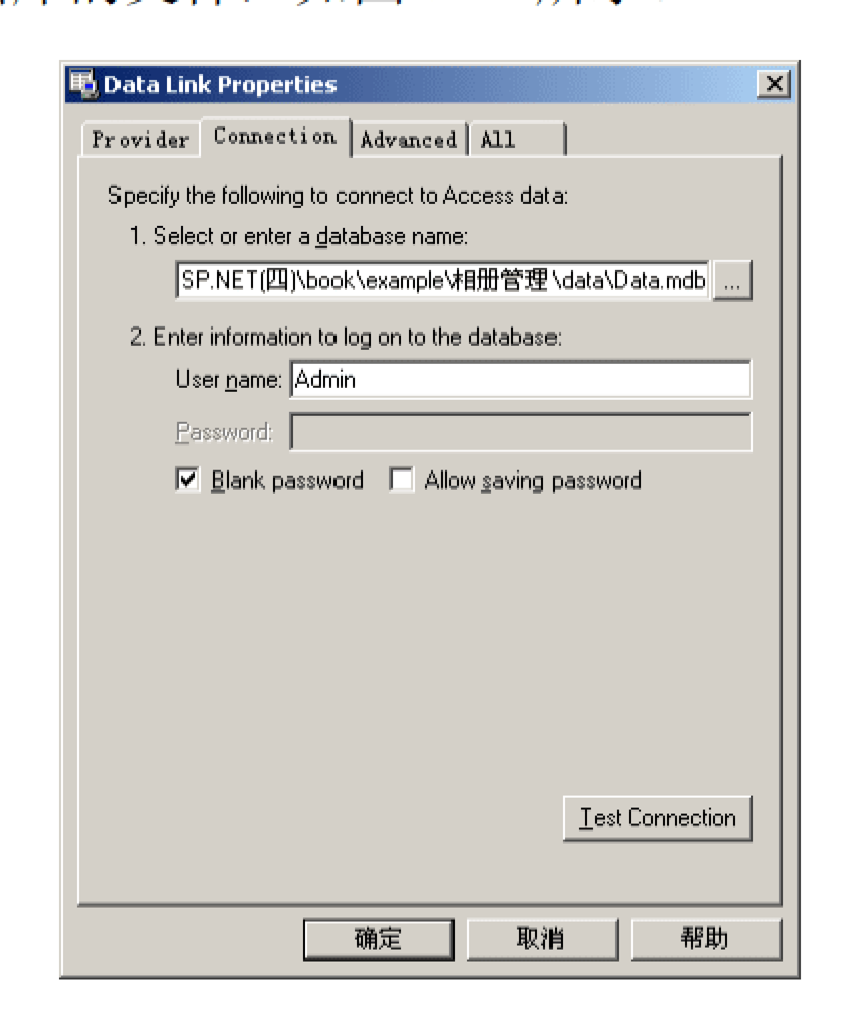


图 9.14　选择数据库

(3) 单击【确定】按钮，返回【OLE DB 连接】对话框，并再次单击【确定】按钮，完成数据库的创建。

3. 创建两个数据集

下面，创建两个数据集，分别用于绑定相片分类和指定分类下相片浏览的显示。

(1) 在【应用程序】面板组中，分别切换至【服务器行为】面板，单击 + 按钮，从弹出的菜单中选择【数据集】命令，在弹出的【数据集】对话框中，设置数据集名称为 DataSet1，连接选择 Cnn，表格选择 PhotoInfo，设置筛选条件为字段 ClassID 等于 URL 变量 ClassID，如图 9.15 所示。

这里所创建的数据集，主要用于绑定指定分类下的相片信息的显示。由于所要显示的相片信息，除了包括 PhotoInfo 数据表中的信息之外，还包括从数据表 PhotoLY 中统计相应相片的留言数，因此仅用一个简单的 Select 语句从数据表 PhotoInfo 中查询信息是不行的。

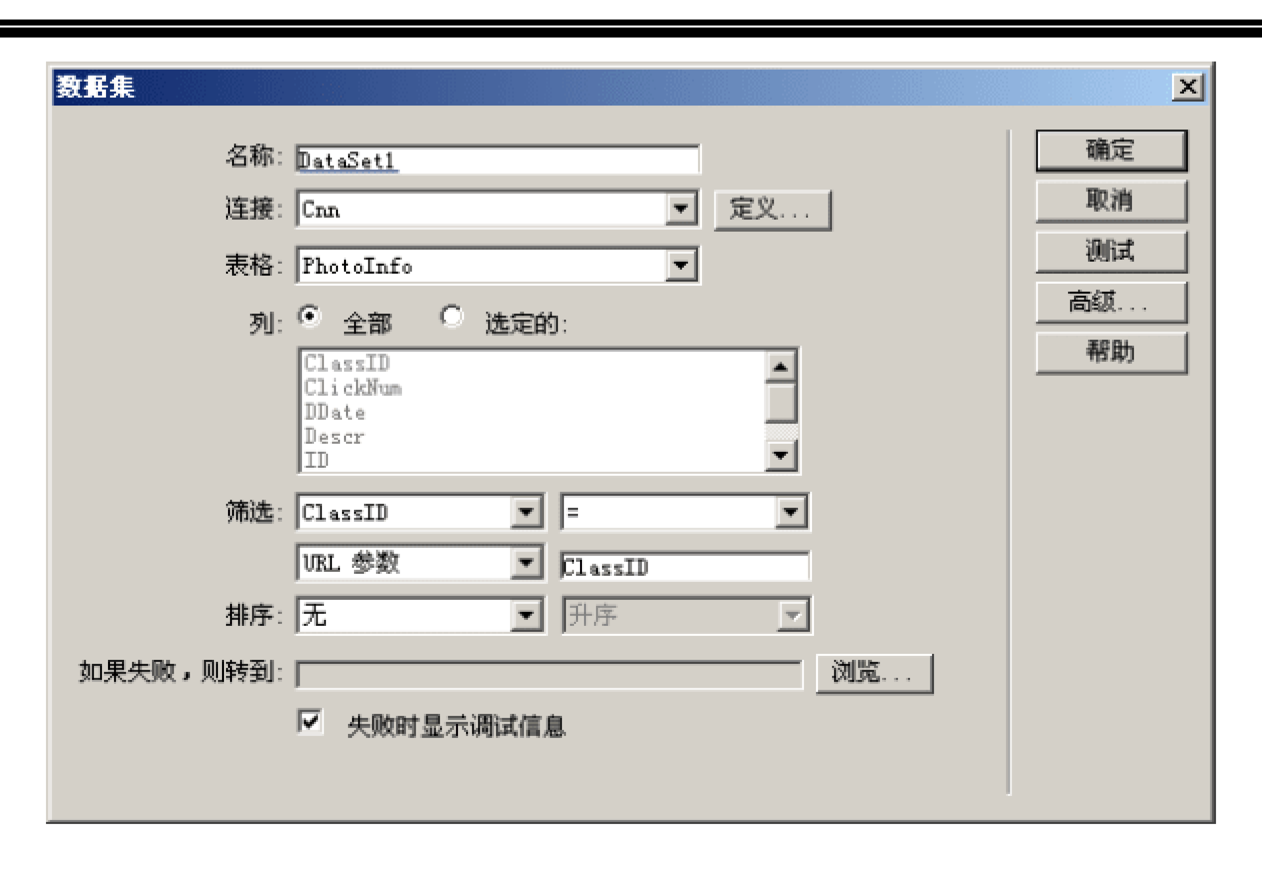


图 9.15　创建数据集 DataSet1

(2) 单击【高级】按钮，此时将弹出【数据集】对话框的高级模式，如图 9.16 所示。

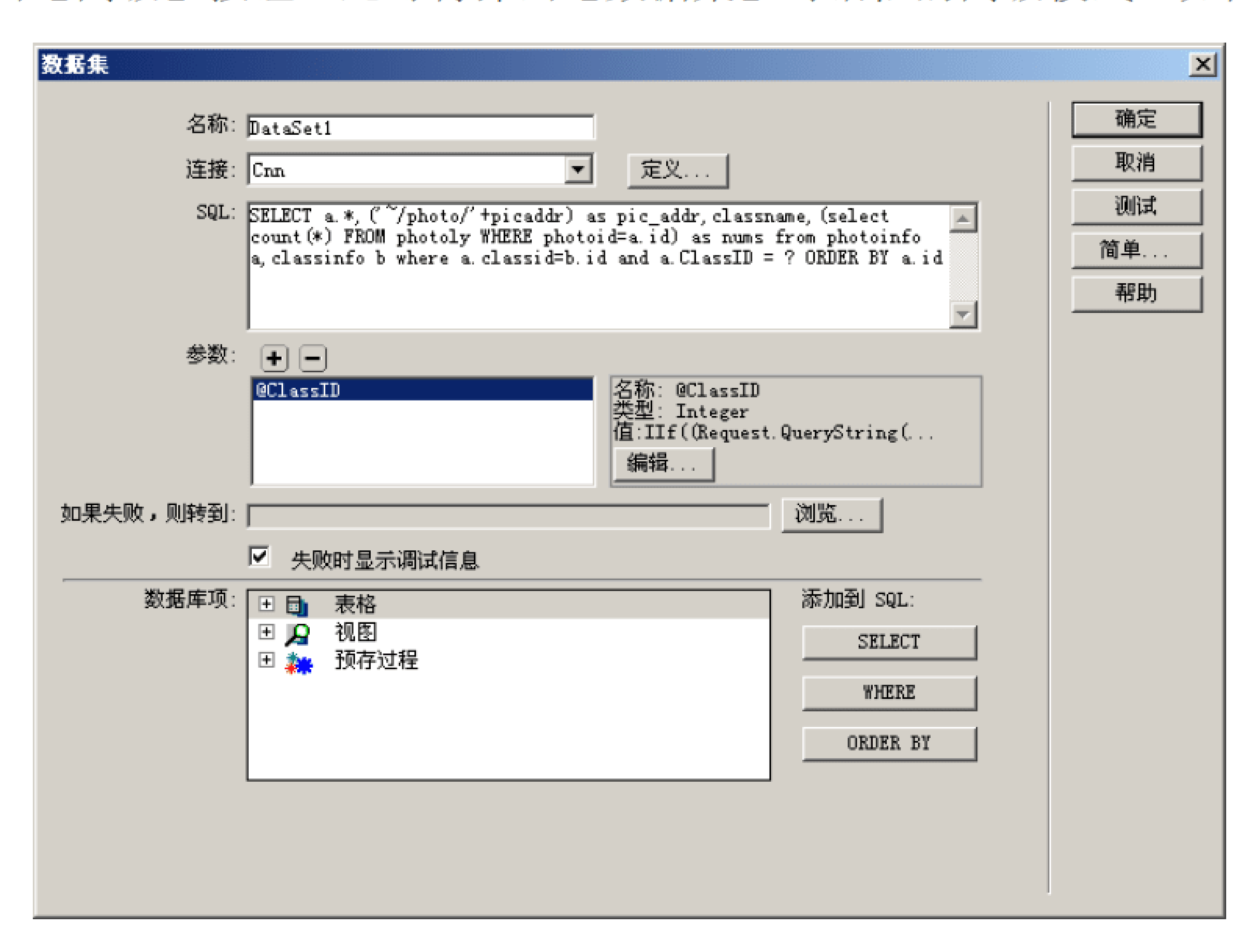


图 9.16　【数据集】对话框的高级模式

在 SQL 后的文本框中，对 SQL 语句进行修改，修改后的语句如下：

```
SELECT a.*,('~/photo/'+picaddr) as pic_addr,classname,(select count(*) FROM
photoly WHERE photoid=a.id) as Nums from photoinfo a,classinfo b where
a.classid=b.id and a.ClassID = ? ORDER BY a.id
```

在此语句中，“?”表示筛选条件中所定义的 URL 参数 ClassID。在所返回的记录中，pic_addr 表示相片地址的完整路径，而 Nums 则表示该相片的留言数。

单击【确定】按钮，完成数据集 DataSet1 的创建。

(3) 在【服务器行为】面板中，再次单击按钮，在弹出的菜单中选择【数据集】命令，新建数据集 DataSet2，如图 9.17 所示。

这里所创建的数据集 DataSet2 用于绑定相片分类的显示。

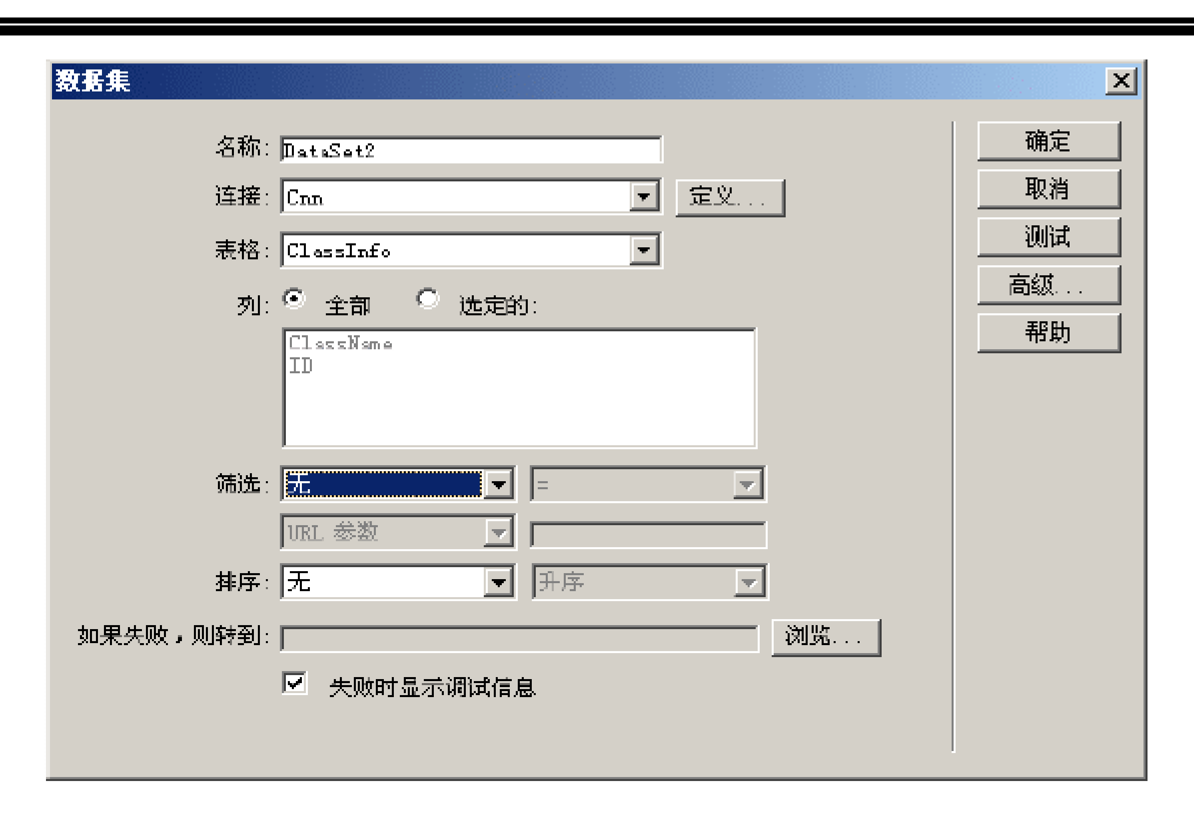


图 9.17　创建数据集 DataSet2

4. 相片分类的绑定显示

添加相片分类的绑定显示具体操作如下。

(1) 将光标置于文本“相片分类”下，在【常用-插入】工具栏中单击按钮，插入一个 <div> 标签。然后切换至【代码】视图，在 <div> 标签中添加样式“style="padding-top:10px"”，如图 9.18 所示。

```
<td width="180" colspan=2 align=center valign="top" bgcolor=
"#FFFFFF" style="padding-top:5px;padding-left:5px">
<span style="font-size:14px"><b>相片分类</b></span>
<div style="padding-top:10px;"></div>
</td>
```

图 9.18　添加标签<div>

(2) 在【应用程序】面板组中切换至【绑定】面板，展开“数据集(DataSet2)”，如图 9.19 所示。

(3) 拖动字段 ClassName 至前面所插入的<div>起始标签和结束标签之间，同时在【属性】工具栏中设置该动态文本的链接，链接地址为：

```
default.aspx?classid=<%# DataSet2.FieldValue("ID", Container) %>
```

即指向本页面，传递参数为相片分类所对应的 ID。

(4) 选择<div>标签，在【应用程序】面板组中，切换至【服务器行为】面板，单击按钮，在弹出的菜单中选择【重复区域】命令，在弹出的【重复区域】对话框中，设置数据集为 DataSet2，并设置显示所有记录，如图 9.20 所示。

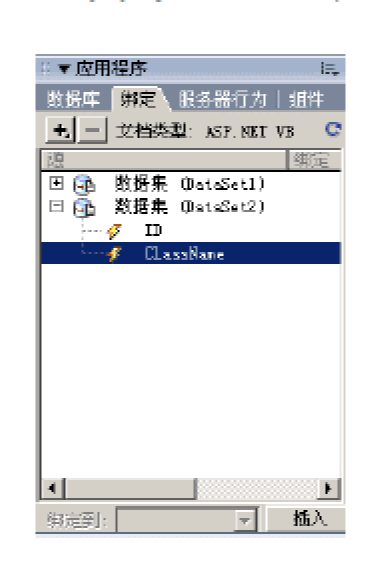


图 9.19　【绑定】面板

图 9.20　【重复区域】对话框

(5) 单击【确定】按钮，完成重复区域的创建。

至此，相片分类的显示绑定完成。

对于指定分类下的相片浏览，其数据的绑定显示比相片分类的绑定更为复杂。因为它所要重复显示的数据并不是一个单独的动态文本，而是一个表格。

5. 创建单条数据的显示

(1) 在相片浏览区域，插入一个 5 行 2 列的表格，修改其代码如下：

【示例代码】

```
<table width="200" border="0" height="220" cellpadding=0 cellspacing=0>
<tr height=130>
  <td colspan=2>
  <div style="width:190px;height:130px; padding-top:5px;background-image:
url('image/photoback.gif');" align=center>
  <a href="ViewPhoto.aspx?id <%# DataSet1.FieldValue("ID", Container) %>"
target=_self>
  <asp:Image ID="Image1" width="180" Height="120" BorderWidth=0  ImageUrl
<%# DataSet1.FieldValue("Pic_Addr", Container) %> runat="server" />
  </a></div>
  </td>
</tr>
<tr height=20>
  <td width="70" align=right>相片名称：</td>
  <td width="130"><font color="red"> <%# DataSet1.FieldValue("PicName",
Container) %> </font></td>
</tr>
<tr height=20>
  <td align=right>点击次数：</td>
  <td><font color="red"> <%# DataSet1.FieldValue("ClickName",
Container) %></font></td>
</tr>
<tr height=20>
  <td align=right>留言次数：</td>
  <td><font color="red"> <%# DataSet1.FieldValue("Nums",
Container) %></font></td>
</tr>
<tr>
  <td align=right valign=top>相片描述：</td>
  <td valign=top><font color="red"> <%# DataSet1.FieldValue("Descr",
Container) %></font></td>
</tr>
</table>
```

在此表格中，第 1 行用于显示相片的图像。这里，插入了一个<div>标签，并设置其背景为 Image 文件夹下的 PhotoBack.gif，其目的是为所显示的图像添加一个边框，以使页面更加美观。表格的其余 4 行分别用于显示相片名称、点击次数、留言次数和相片描述，其显示信息分别从数据集 DataSet1 中的相应字段中获取。

(2) 将以上显示单条数据的表格代码剪切(注意，是剪切而不是复制)。打开【应用程序】面板组，切换至【服务器行为】面板，单击按钮，从弹出的菜单中选择【数据列表】

命令，在弹出的【数据列表】对话框中，设置其 ID 为 DataList1，数据集选择 DataSet1，在【显示】单选按钮组中选择【所有记录】单选按钮。选择模板列表中的【项目】选项，在内容中粘贴前面剪切的内容，然后在【组织项】单选按钮组中选择【使用表】单选按钮，并在【表列】下拉列表框中选择 3，在【表格单元格填充顺序】单选按钮组中选择【自上而下换行】单选按钮，如图 9.21 所示。

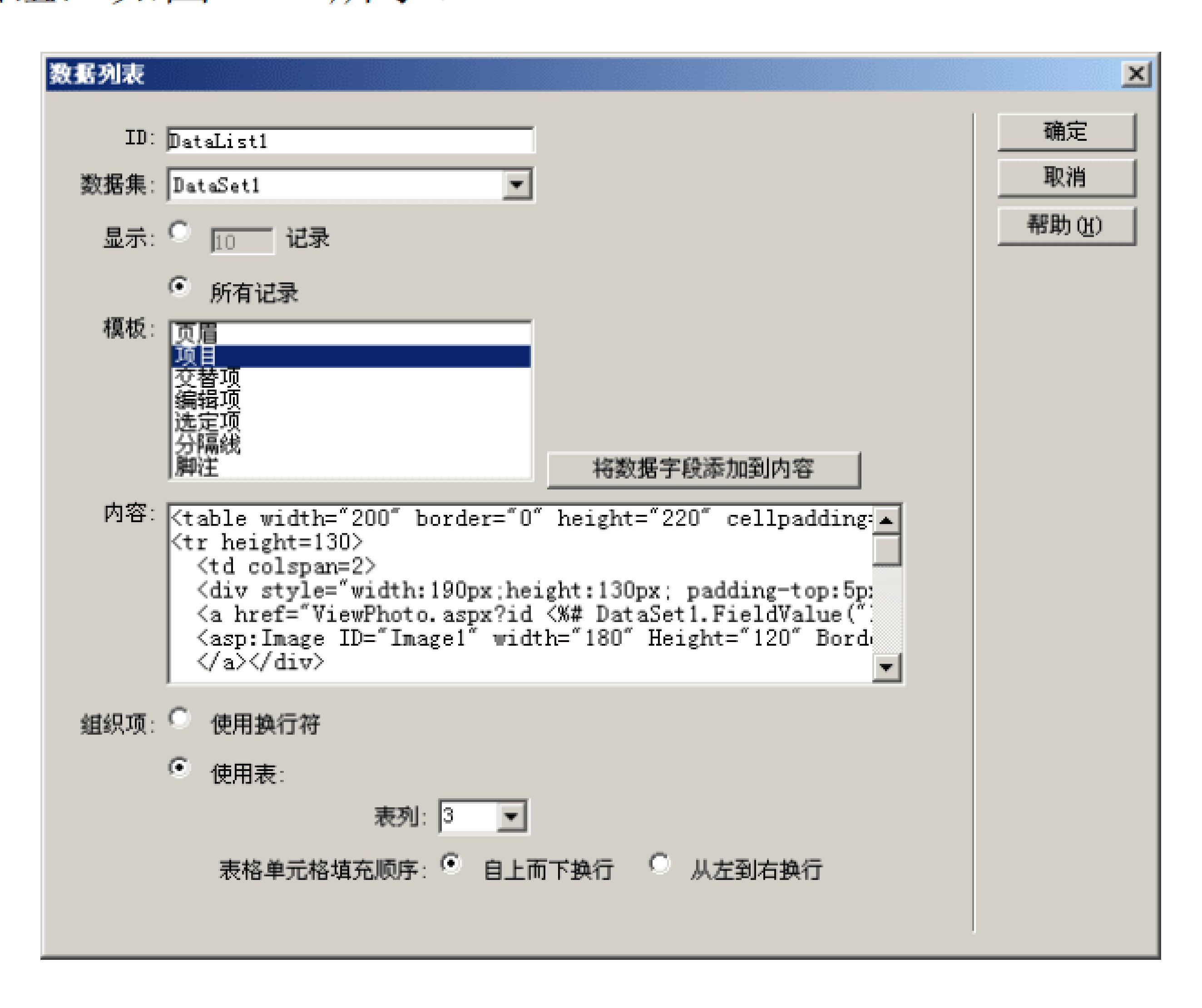


图 9.21　【数据列表】对话框

(3) 单击【确定】按钮，完成数据列表的添加。下一步，再来设置重复显示项目的对齐方式。在【代码】视图中，将光标置于<asp:DataList>标签中的 ItemTemplate 项的结束标签之后，单击【ASP.NET-插入】工具栏中的【更多标签】按钮，在弹出的【标签选择器】对话框中，选择【ASP.NET 标签】|【样式】分类中的 ItemStyle，如图 9.22 所示。

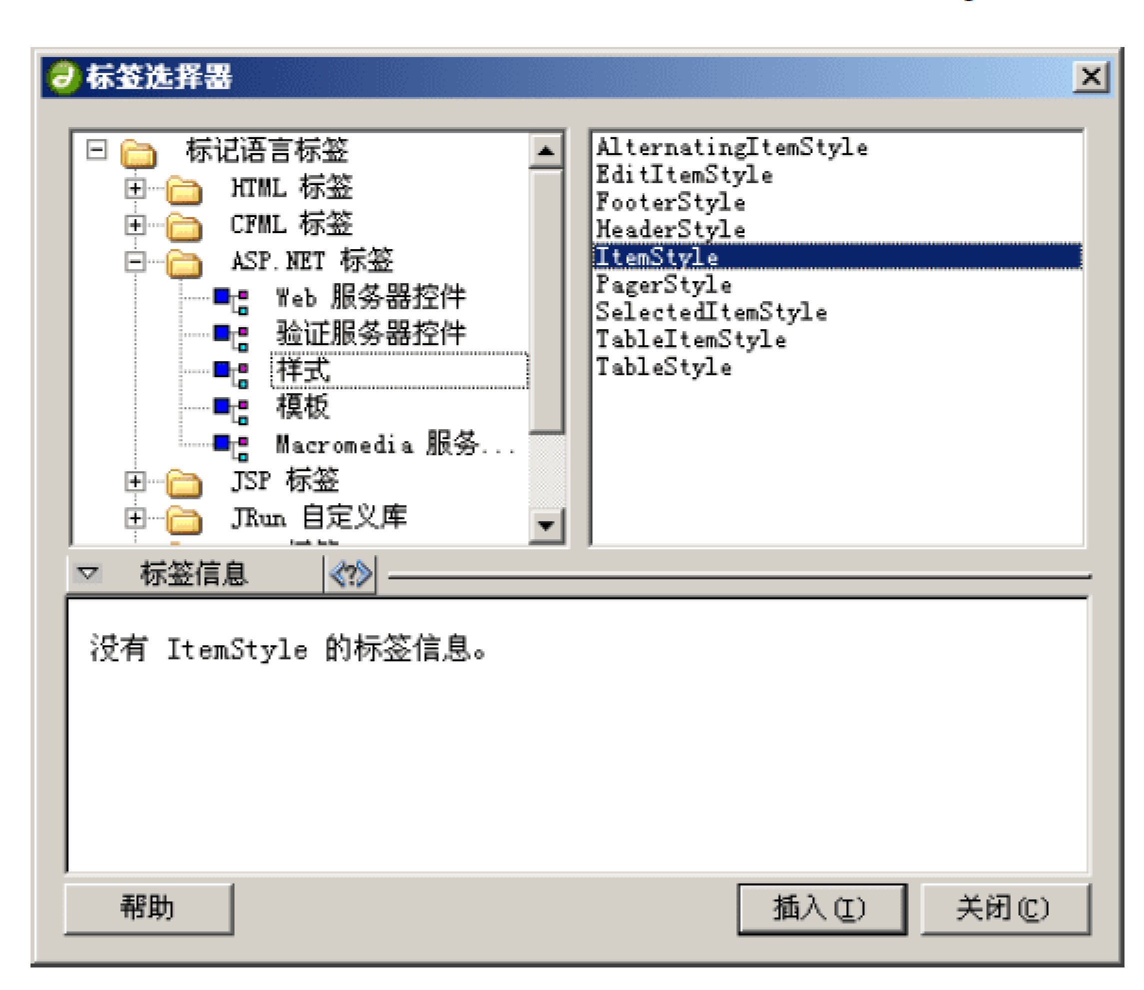


图 9.22　【标签选择器】对话框

(4) 单击【插入】按钮，在弹出的【标签编辑器】对话框中，将【垂直对齐方式】选项设置为【顶端】，如图 9.23 所示。

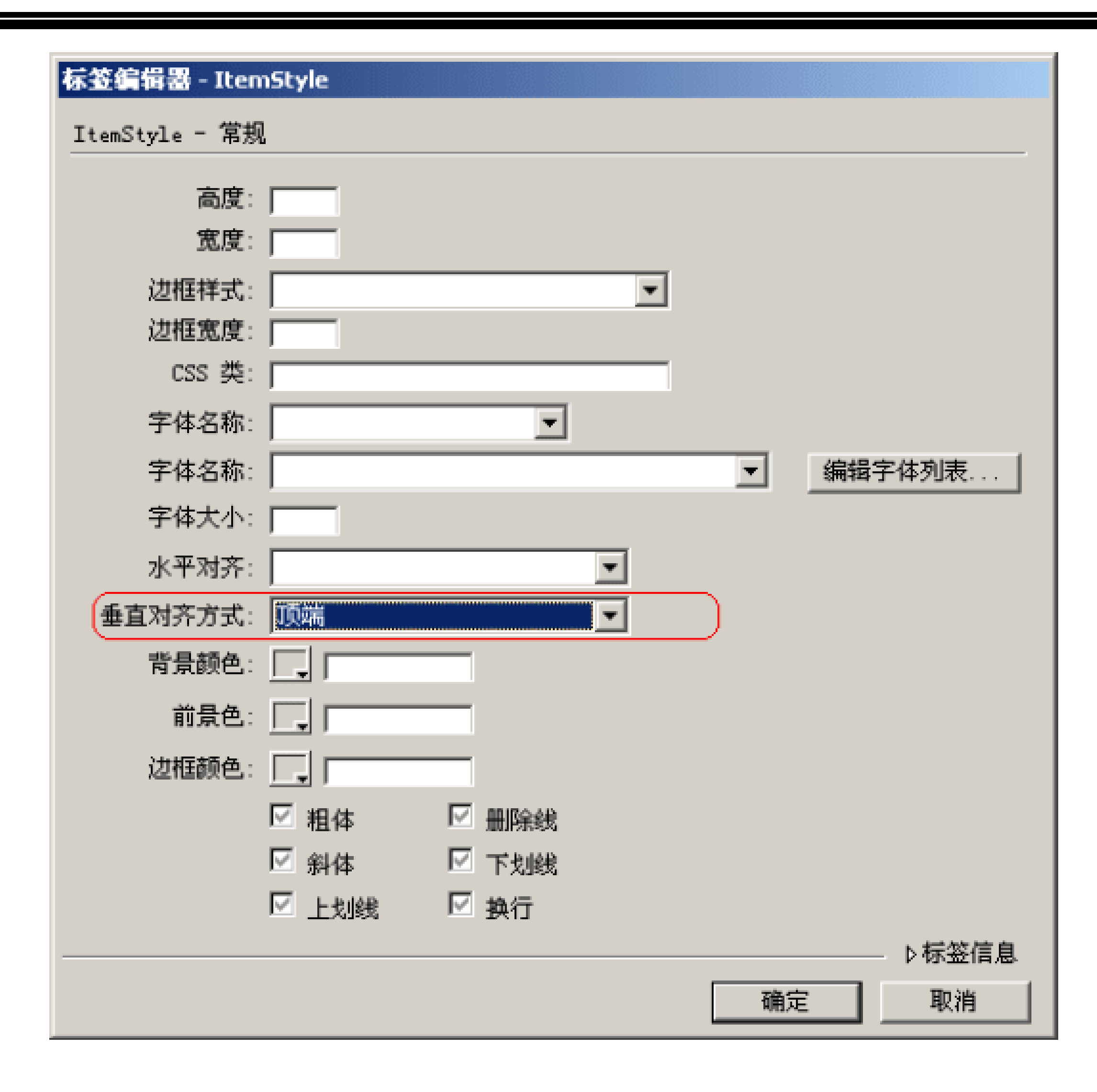


图 9.23　【标签编辑器】对话框

(5) 单击【确定】按钮，此时在代码中将会插入 DataList 控件中的 itemstyle 项，如图 9.24 所示。

```
        </table>
      </ItemTemplate>
    <itemstyle VerticalAlign="Top"></itemstyle>
    </asp:DataList>
  </center>
```

图 9.24　添加 ItemStyle 项

至此，指定分类下的相片浏览数据绑定就完成了。

从查看页面的基本结构(如图 9.12 所示)中可以发现还有一个动态数据需要显示，那就是文本“当前分类”后所要显示的当前分类名称。虽然在数据集 DataSet1 中可以返回当前所浏览的相片的分类名称，但是当指定分类下没有相片时，此数据为空。为此，可以再次创建一个数据集 DataSet3，其属性设置如图 9.25 所示。

图 9.25　添加数据集 DataSet3

这里所选择的表格为 ClassInfo，筛选条件设置为字段 ID 等于 URL 参数 ClassID。在创

建数据集 DataSet3 之后，打开【应用程序】面板组，选择【绑定】面板，在显示出来的数据集中选择 DateSet3，并将其展开，如图 9.26 所示。

单击字段 ClassName，并将其拖至文本“当前分类”之后，在【属性】面板中设置其文本显示颜色为红色。

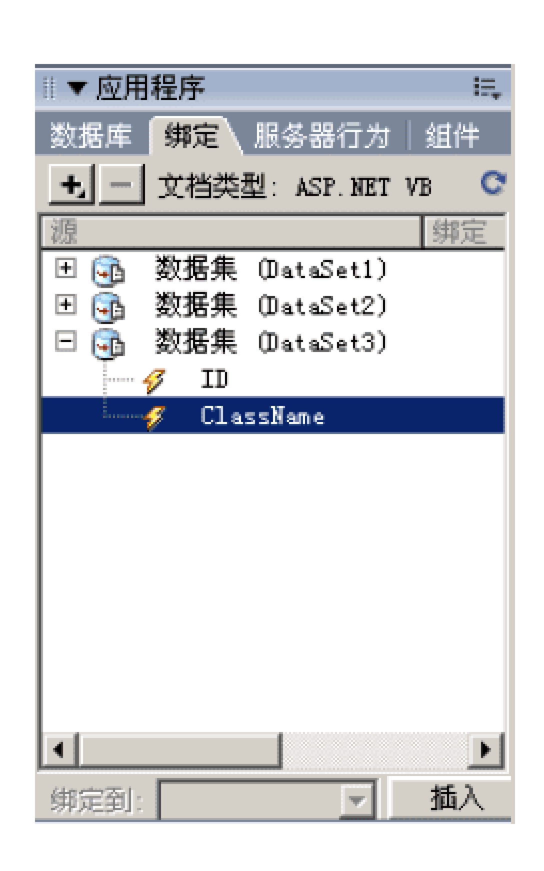


图 9.26　【绑定】面板

至此，页面的功能实现基本完成。细心的读者也许会发现，对于数据集 DataSet1 和 DataSet3，其筛选的条件变量均为 URL 变量 ClassID，即页面所传递的参数 ClassID。而当用户浏览网络相册的首页面时，是不可能带参数的，此时数据集 DataSet1 和 DataSet3 所返回的记录均为空，以致于页面无法正常显示数据。例如，用户在地址栏中输入“http://localhost/wlxc/default.aspx”来访问网络相册，此时没有指定相片分类，结果就无法显示相应的相片；而用户在地址栏中输入“http://localhost/wlxc/default.aspx?ClassID=1”来访问网络相册，则可显示分类 ID 为 1 的所有相片，但不可能要求用户按指定的标准来输入地址进行访问，这也是不现实的。虽然这并不会导致页面的异常发生，但是初次打开页面，什么也看不到，只有选择分类后才能看到相关相片，这会让用户觉得有点不舒服。

那么，有什么办法可以解决以上问题呢？很简单，可以在页面的 Page_Load 事件中添加代码，对页面参数 ClassID 是否为空进行判断。如果为空，则将页面再次跳转至本页面，并将数据表 ClassInfo 中的第一个分类 ID(可从数据集 DataSet2 中的第一条记录获取)作为页面参数进行传递。也就是说，当用户访问首页面时，默认显示第一个分类的所有相片。

Page_Load 事件的代码如下：

【示例代码】

```
Sub Page_Load(ByVal Sender As Object, ByVal E As EventArgs)
  If Not IsPostBack() Then
    If Request.QueryString("ClassID")="" then
      '当页面参数 ClassID 为空时
      If DataSet2.DefaultView.Table.Rows.Count>0 Then
      '当数据集 DataSet2 存在记录时，获取其首条记录的字段 ID 的值，并作为页面参数传递
        Response.Redirect("Default.aspx?ClassID=" & DataSet2.DefaultView.
Table.Rows(0)("ID"))
      End If
    End If
  End If
End Sub
```

至此，页面的功能实现全部完成，页面预览效果如图 9.27 所示。

点击不同的分类，可浏览该分类下的所有相片。点击某个相片，则可查看该相片的相关留言，同时也可发表留言，这也是接下来所要介绍的内容。

图 9.27　页面预览效果

9.2.3　查看相片

在查看相片的页面中，其主要功能可分为以下 4 个部分。

- 显示指定相片的基本信息，包括相片名称、上传时间、点击次数、留言次数和相片描述等；
- 显示指定相片所属分类下的其他相关相片；
- 显示指定相片的留言信息；
- 允许发表留言。

下面介绍该页面的具体实现。

新建一个 ASP.NET VB 类型的页面，将其命名为 ViewPhoto.aspx。选择【修改】|【模板】|【套用模板到页】命令，在弹出的【选择模板】对话框中选择【前台模板】，单击【选定】按钮，将模板应用到页面。

在实现页面之前，先来考虑页面的功能区域的划分。对于相关相片的显示，可在可编辑区域 EditRegion3 中实现，而其余功能则可在可编辑区域 EditRegion4 中实现，如图 9.28 所示。

接下来，将逐步实现各项功能。

首先，来看相关相片的显示。所谓相关相片，是指与所查看的相片同一类的其他相片。其数据的显示可通过“重复区域”来实现，为此先来创建一个数据集。

(1) 在【应用程序】面板组中，切换至【服务器行为】面板，单击 ⊞ 按钮，从弹出的菜单中选择【数据集】命令。在弹出的【数据集】对话框中，单击【高级】按钮，切换至【数据集】对话框的高级模式。单击【添加参数】按钮 ＋，在弹出的【编辑参数】对话框

中，设置名称为“@ID”，类型选择 BigInt，如图 9.29 所示。

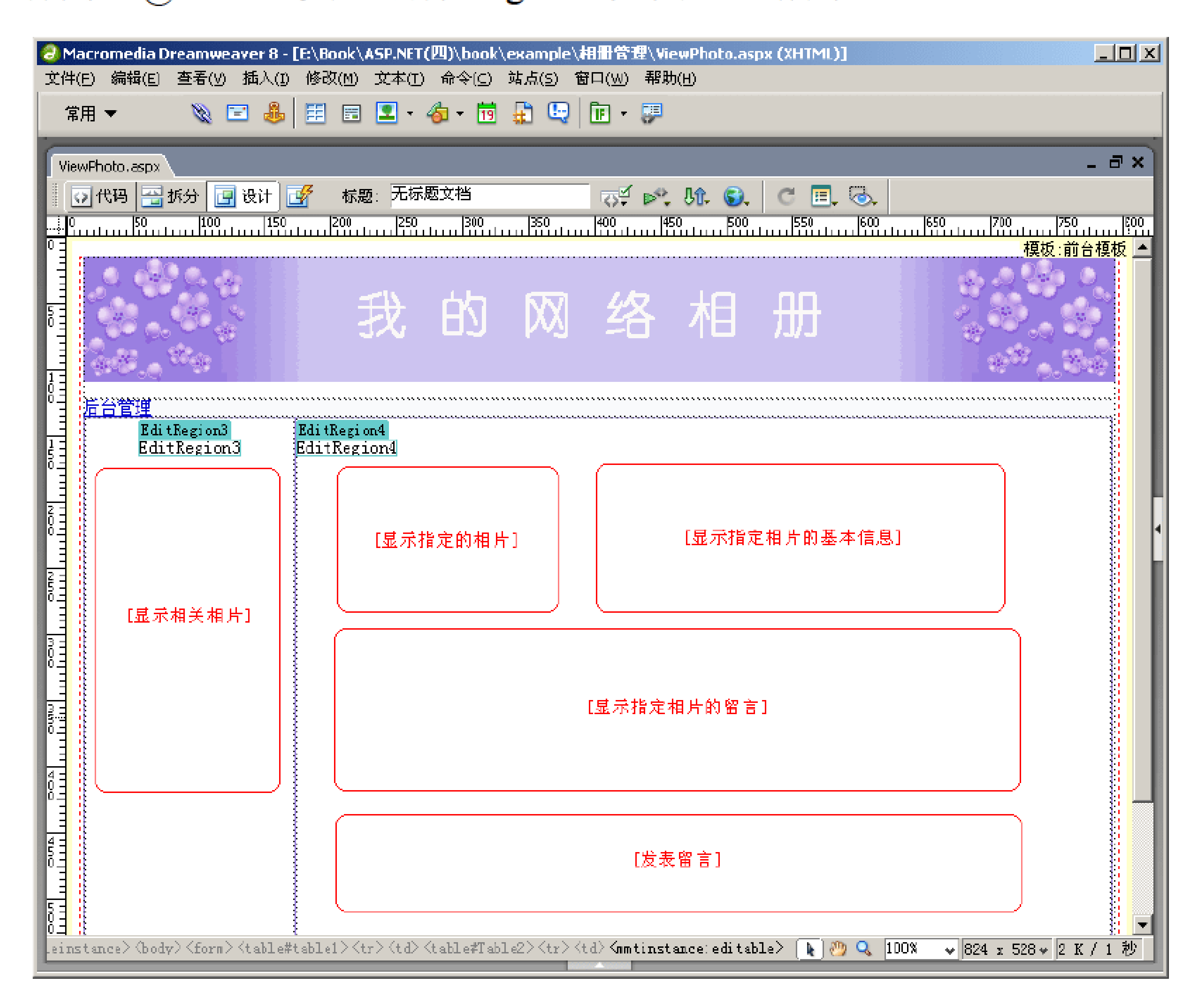


图 9.28　页面功能区域分布

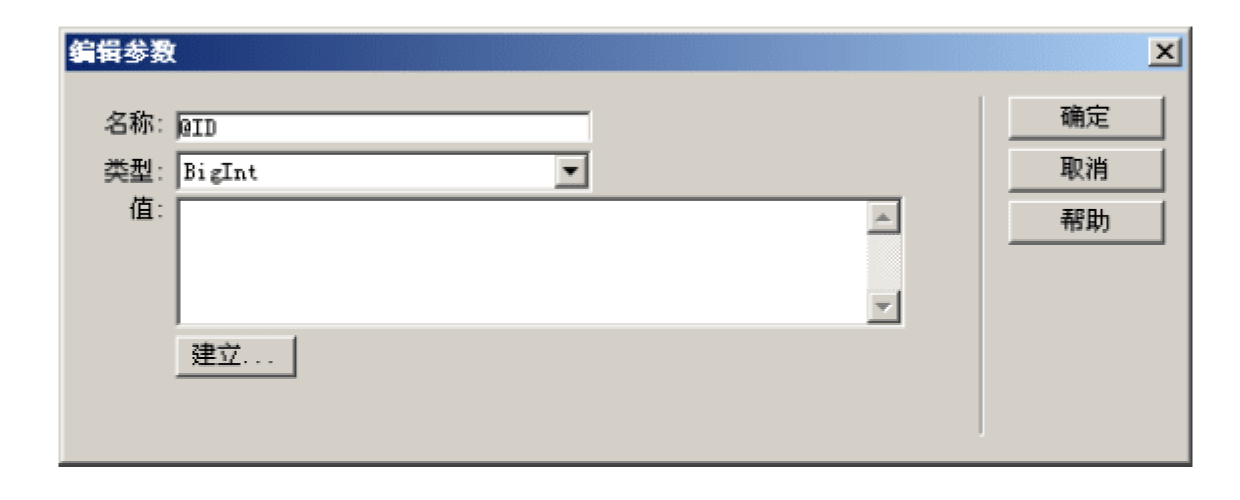


图 9.29　【编辑参数】对话框

(2) 单击【建立】按钮，在弹出的【生成值】对话框中，设置【名称】为 ID，【源】为【URL 参数】，如图 9.30 所示。

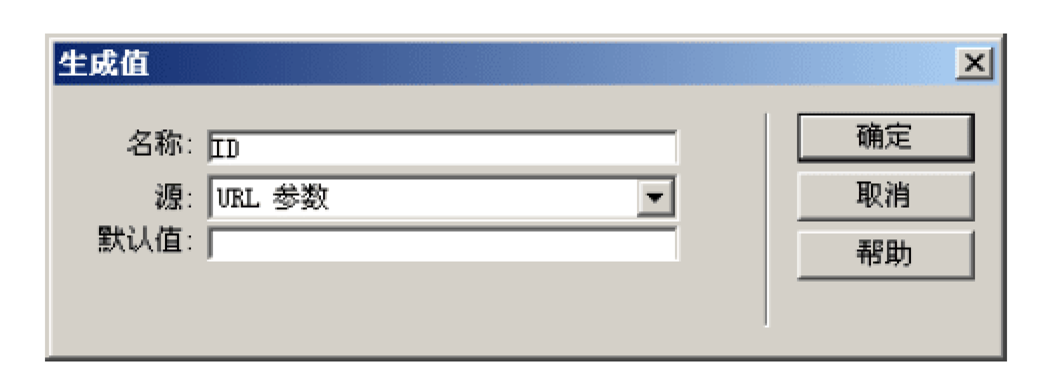


图 9.30　【生成值】对话框

(3) 单击【确定】按钮，返回【编辑参数】对话框，再次单击【确定】按钮，完成此参数的添加。由于在 SQL 语句中，将会两次调用到此参数，因此需要重复以上步骤，添加一个名为“@ID1”的参数，其取值与参数“@ID”的取值一样。

(4) 在 SQL 文本框中输入用于查询数据的 SQL 语句，如图 9.31 所示。

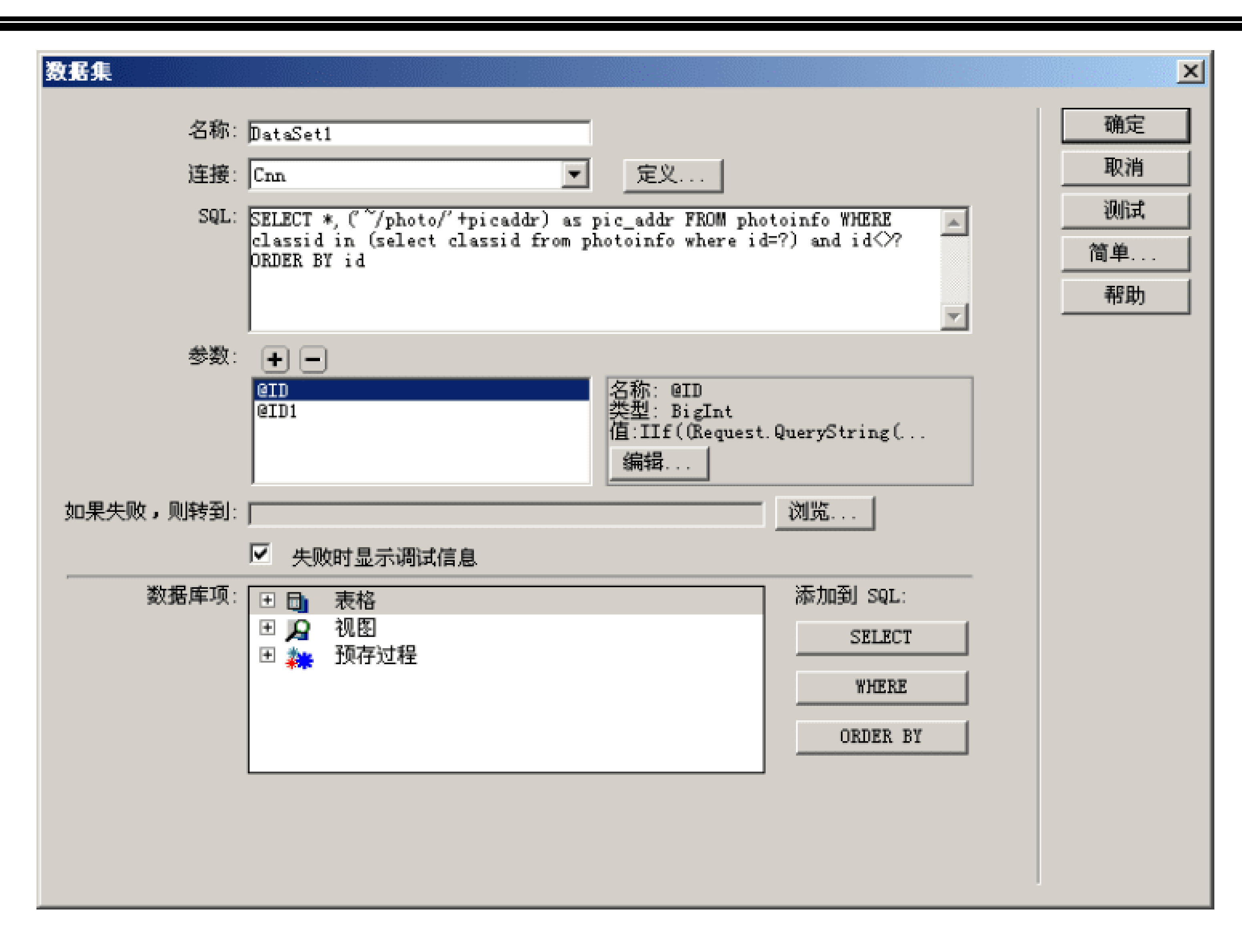


图 9.31　创建数据集 DataSet1

在此 SQL 语句中，使用了一个 Select 子查询来获取指定相片 ID 所属的相片分类。单击【确定】按钮，完成数据集的创建。

(5) 将光标移至 EditRegion3 可编辑区域中，添加文本“相关相片”，设置其字体大小为 14 像素，并加粗显示。

(6) 在文本“相关相片”下，单击【常用-插入】工具栏中的按钮，插入一个<div>标签，用于控制单条数据的显示。编辑<div>标签，添加 style 样式属性 padding-bottom，设置其值为 5 像素。将光标置于<div>标签中，单击【ASP.NET-插入】工具栏中的【更多标签】按钮，在弹出的【标签选择器】对话框中，选择【ASP.NET 标签】|【Web 服务器控件】分类中的 asp:Image，插入一个图像服务器控件，设置其 ImageUrl 属性的值为<%# DataSet1.FieldValue("pic_addr", Container) %>，Width 属性为 140。同时，在【属性】面板中设置其链接属性为 Viewphoto.aspx?id=<%# DataSet1.FieldValue("ID", Container) %>，链接目标为_self，如图 9.32 所示。

```
            <div style="padding-bottom:5px;">
            <a href="viewphoto.aspx?id=<%# DataSet1.FieldValue("ID", Container) %>"
target="_self"><asp:Image ImageUrl=<%# DataSet1.FieldValue("pic_addr", Container) %>
Width="140" runat="server" /></a>
            </div>
```

图 9.32　插入图像

(7) 在所创建的图像链接之后，添加标签
，以便换行。然后，打开【应用程序】面板组，选择【绑定】面板，展开数据集 DataSet1，选择其下面的字段 PicName，并将其拖至前面所创建的图像链接之下，以绑定显示相应相片的名称。这样，即完成了单条数据的绑定。

(8) 选定整个<div>标签，在【应用程序】面板组中，切换至【服务器行为】面板，单击按钮，从弹出的菜单中选择【重复区域】命令，在弹出的【重复区域】对话框中，选择 DataSet1 作为重复区域的数据集，并设置所有记录如图 9.33 所示。

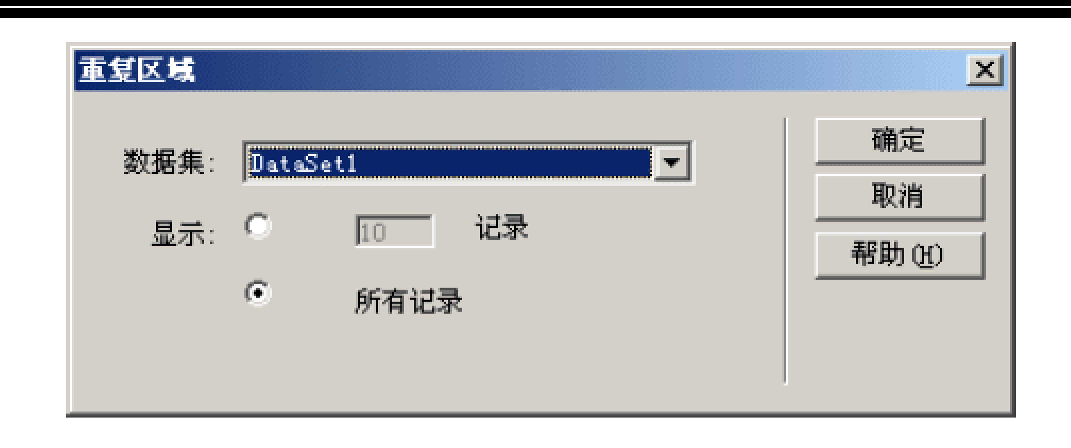


图 9.33　【重复区域】对话框

单击【确定】按钮，完成重复区域的设置。此时，对于相关相片的数据绑定全部完成。接下来，实现相片信息的数据绑定显示。

(9) 将光标移至 EditRegion4 可编辑区域中，插入一个 6 行 3 列的表格，合并第 1 列的 6 个单元格，并在合并的单元格中插入一个 Image 服务器控件，在其他单元格中输入相应的文本，如图 9.34 所示。

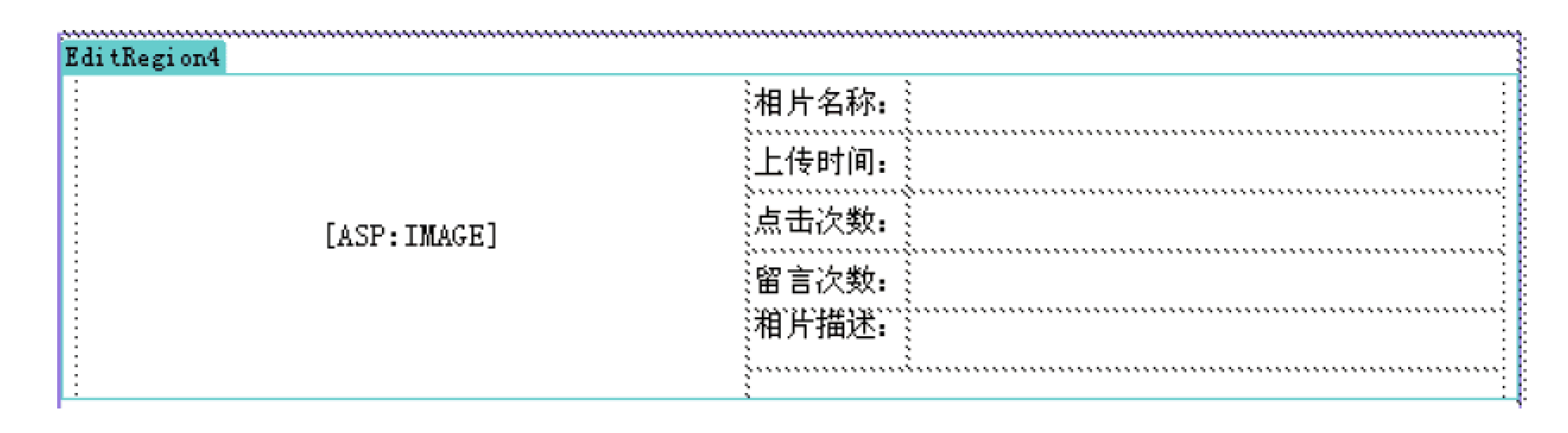


图 9.34　插入的表格

(10) 为了绑定相片数据的显示，这里需要创建一个新的数据集。

在【应用程序】面板组中，切换至【服务器行为】面板，单击按钮，从弹出的菜单中选择【数据集】命令。在弹出的【数据集】对话框中，设置数据集名称为 DataSet2，连接选择 Cnn，表格选择 PhotoInfo，筛选条件设置为字段 ID 等于 URL 参数 ID，如图 9.35 所示。

图 9.35　创建数据集 DataSet2

(11) 为了在返回的数据中包含指定相片文件的完整路径和统计其留言数，需要对 SQL 语句进行调整。单击【高级】按钮，切换至【数据集】对话框的高级模式。在数据集的高级模式对话框中，修改 SQL 语句为：

```
SELECT a.*,('~/photo/'+picaddr) as Pic_Addr,(select count(*) FROM PhotoLy
WHERE photoid=a.id) as Nums from photoinfo a where a.id=?
```

这里，所返回的字段 Pic_Addr 为指定相片所对应的图像文件的完整路径，而字段 Nums 则为指定相片的留言数。单击【确定】按钮，完成数据集的创建。

(12) 返回刚才在 EditRegion4 可编辑区域所添加的表格，设置 Image 控件的 ImgUrl 属性为<%# DataSet2.FieldValue("pic_addr", Container) %>，Width 属性为 250。

(13) 在【应用程序】面板组中，切换至【绑定】面板，选择数据集 DataSet2，并将其展开。拖动字段 PicName 至文本“相片名称”后的单元格中，并在【属性】工具栏中设置该动态文本的字体颜色值为“#3399CC”，加粗显示。与此类似，依次拖动字段 Ddate、ClickNum、Nums 和 Descr 至文本“上传时间”、“点击次数”、“留言次数”、“相片描述”后的单元格中，并为其设置相应的字体。至此，指定相片信息的数据绑定完成。

接下来，实现留言信息的数据绑定显示。

(14) 在【应用程序】面板组中，将选项切换至【服务器行为】面板，单击按钮，从弹出的菜单中选择【数据集】命令。在弹出的【数据集】对话框中，设置数据集名称为 DataSet3，连接选择 Cnn，表格选择 PhotoLY，筛选条件设置为字段 PhotoID 等于 URL 参数 ID，如图 9.36 所示。

图 9.36　创建数据集 DataSet3

(15) 单击【确定】按钮，完成数据集的创建。这里创建的数据集所返回的记录为指定相片的所有留言信息。

下面，先来创建单条留言信息的数据绑定显示。

(16) 在 EditRegion4 可编辑区域中，再次插入一个 2 行 2 列的表格，将第 1 行的背景颜色值设置为“#CCC3F0”，字段颜色值设置为“#FFFFFF”，然后在第 1 行的第 1 个单元格中输入文本“留言人：”，第 2 个单元格中输入“留言时间：”，并将第 2 个单元格的水平对齐方式设置为“右对齐”，最后再合并第 2 行的两个单元格，如图 9.37 所示。

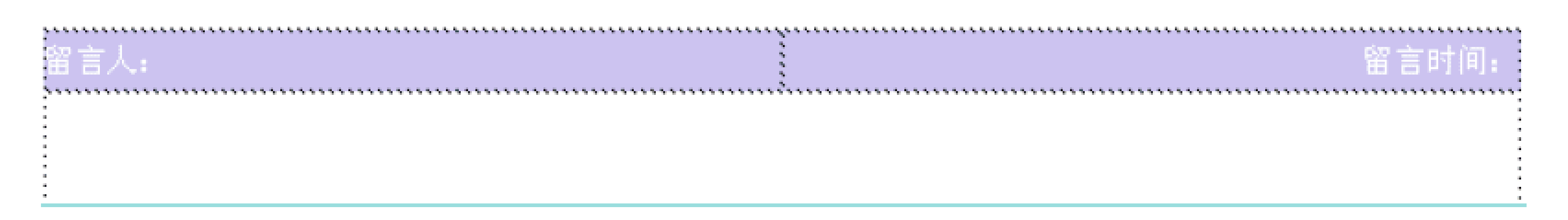


图 9.37　插入表格

(17) 在【应用程序】面板组中，切换至【绑定】面板，选择数据集 DataSet3，并将其展开，如图 9.38 所示。

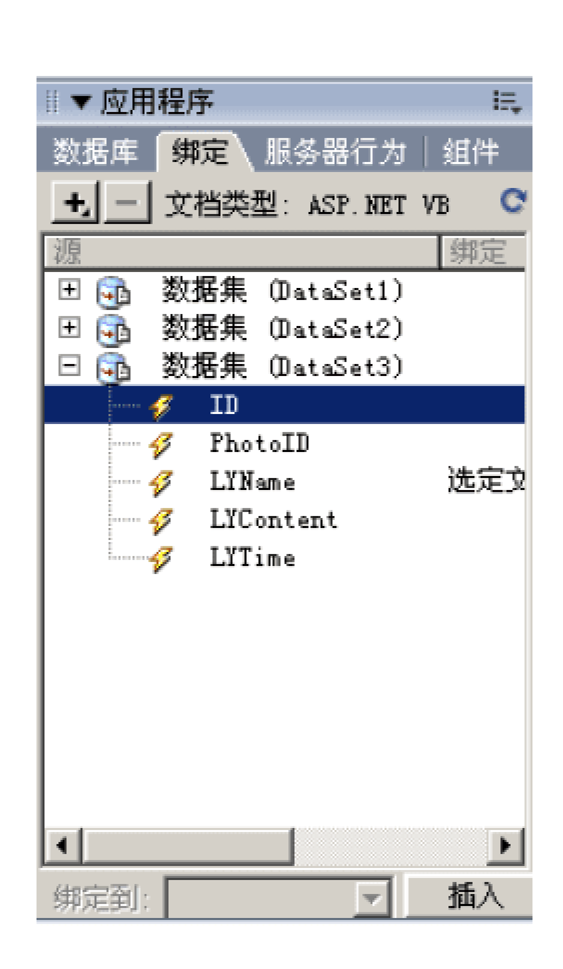


图 9.38　【绑定】面板

(18) 选择字段 LYName 并将其拖动至文本“留言人：”之后，选择并拖动字段 LYTime 至文本“留言时间：”之后，选择并拖动字段 LYContent 至表格第 2 行的单元格中。在【应用程序】面板组中，切换至【服务器行为】面板，其中显示了刚才所绑定的动态文本。双击“动态文本(DataSet3.LYContent)”，在弹出的【动态文本】对话框中，设置其格式为【编码—HTML 编码格式】，如图 9.39 所示。

单击【确定】按钮，完成单条留言数据的绑定显示。

(19) 选定显示单条留言信息的整个表格，在【应用程序】面板组的【服务器行为】面板中，单击 + 按钮，从弹出的菜单中选择【重复区域】命令，在弹出的【重复区域】对话框中，选择 DataSet3 作为数据集，并设置显示所有记录，如图 9.40 所示。

图 9.39　设置动态文本格式

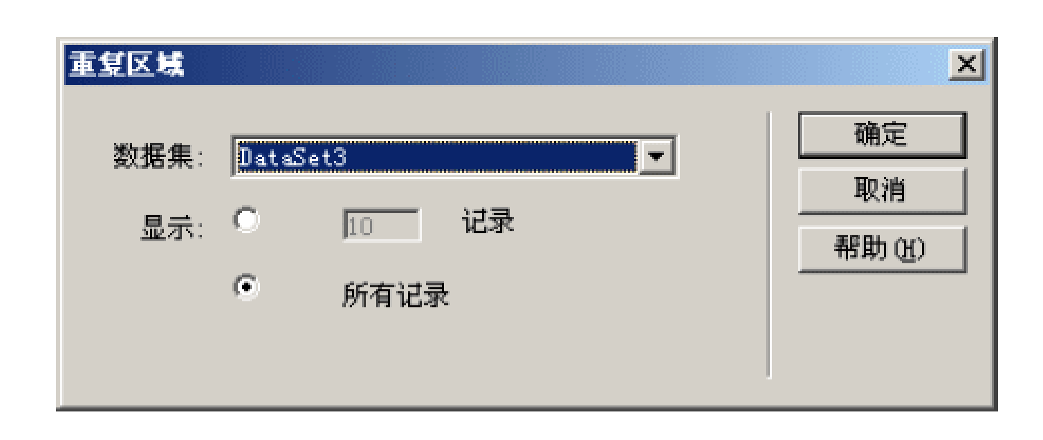


图 9.40　设置重复区域

(20) 单击【确定】按钮，完成重复区域的设置，同时也完成了指定相片的所有留言信息的绑定。

至此，页面仅剩下发表留言的功能尚未完成，该功能将在 9.2.4 节中介绍。此时，页面预览如图 9.41 所示。

图 9.41　页面预览

9.2.4　发表留言

这里，将继续完善用于查看相片的页面 ViewPhoto.aspx，添加发表留言的功能。

在 EditRegion4 可编辑区域中，已经添加了一个用于显示相片信息的表格和一个用于显示留言信息的 Repeater 数据控件(即重复区域)。这里，再插入一个表单，并在【代码】视图中添加其 Runat 属性为 Server。然后，在表单中插入一个 4 行 2 列的表格，表格属性如图 9.42 所示。

图 9.42　表格属性

分别合并表格第 1 行和第 4 行的两个单元格，并在表格中添加相应的文本和控件，如图 9.43 所示。

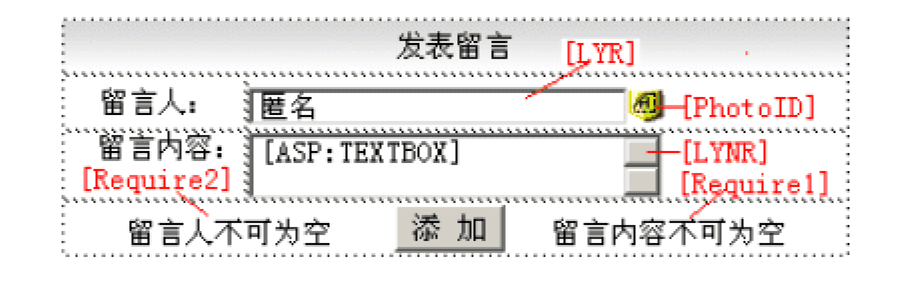


图 9.43　插入表格

图中，红色字体标记了各控件所对应的 ID 名称。其中，文本框控件 LYR 的初始值设

置为“匿名”；文本框控件 LYNR 的文本模式设置为【多行】，行数为 6。控件 Require1 和 Require2 为两个 RequiredFieldValidator 验证服务器控件，分别对文本框 LYNR 和文本框 LYR 的输入进行验证，要求其必须输入。

此外，特别需要提到的是，在表格中添加了一个 ID 为 PhotoID 的隐藏域，其取值设置为<%# DataSet2.FieldValue("ID", Container) %>，即当前相片的 ID，其作用在于，当插入记录时，可以从该隐藏域的取值中获取当前留言所针对的相片 ID。

用于发表留言信息的表单设计完成，下面通过【服务器行为】面板中的【插入记录】命令来完成数据的添加。

在【服务器行为】面板中，单击 + 按钮，从弹出的菜单中选择【插入记录】命令，此时将弹出【插入记录】对话框，如图 9.44 所示。

图 9.44　【插入记录】对话框

在【插入到表格】下拉列表框中，选择数据表 PhotoLY，然后分别设置其各个字段的获取值。对于字段 ID 和字段 LYTime，均存在默认值，此处无须设置。字段 LYName、LYContent 和 PhotoID 的获取值分别设置为表单元素 Lyr、Lynr 和 PhotoID 的取值。如果数据插入成功，则转向首页面 Default.aspx。单击【确定】按钮，完成插入记录的设置。

最后，切换至【代码】视图，添加页面的 Page_Load 事件，该事件主要用于在数据库中对指定相片的点击次数执行加 1 操作，其代码如下：

【示例代码】

```
Sub Page_Load(ByVal Sender As Object, ByVal E As EventArgs)
  Dim StrCnn As String
  Dim StrSql As String
  Dim Cnn As OleDbConnection
  Dim Cmd As OleDbCommand
  If Not IsPostBack() Then
    '判断页面参数 ID 是否为空
    if request.QueryString("id")<>"" then
      '获取数据库连接字符串
      StrCnn =
System.Configuration.ConfigurationSettings.AppSettings("MM_CONNECTION_ST
RING_Cnn")
      Cnn = New OleDbConnection(StrCnn)
      Cnn.Open()
      '更新指定相片的点击次数，其值在原值基础上加 1
```

```
        StrSql = "update photoinfo set clicknum=clicknum+1 where id=" &
    request.QueryString("id")
        Cmd = New OleDbCommand(StrSql, Cnn)
        Cmd.ExecuteNonQuery()
        Cnn.Close()  '关闭数据库连接
      end if
    End If
End Sub
```

整个页面的功能设计已经全部完成，页面预览如图 9.45 所示。

图 9.45　页面预览

当用户发表留言时，也许会发现一个问题，即当在留言人文本框中不输入其他值，而使用默认值“匿名”时，所提交的数据中字段 LyName 的值为空。这是由于在插入记录时，Dreamweaver 没有获取到文本框的默认值。解决办法是，在【代码】视图中找到<MM:Insert>标签中的<Parameters>子项下的参数@LYName 的定义代码，设置 IIF 函数的第 3 个参数(即当表单变量 LYR 的取值为空时所返回的值)为“匿名”，如图 9.46 所示。

```
  <Parameters>
    <Parameter Name="@LYContent" Value='<%# IIf((Request.Form("lynr") <> Nothing), Request.Form(
"lynr"), "") %>' Type="WChar" />
    <Parameter Name="@LYName" Value='<%# IIf((Request.Form("lyr") <> Nothing), Request.Form(
"lyr"), "匿名") %>' Type="WChar" />
    <Parameter Name="@PhotoID" Value='<%# IIf((Request.Form("PhotoID") <> Nothing), Request.Form
("PhotoID"), "") %>' Type="Integer" />
  </Parameters>
</MM:Insert>
```

图 9.46　修改代码

至此，整个网络相册系统的前台功能全部设计完成。

9.3　后台功能实现

在网络相册的后台功能中，主要包括对相片分类、相片信息和相片的留言信息进行管理。

9.3.1　构建模板页

在实现后台功能之前，首先来构建后台页面的模板页。对于后台页面来说，其页面结构与前台页面的结构类似，因此可以在前台模板的基础上进行调整。

打开前台的模板文件“前台模板.dwt.aspx”，选择【文件】|【另存为模板】命令，在弹出的【另存为模板】对话框中，将文件名设置为“后台模板”，如图 9.47 所示。

图 9.47　另存为模板

单击【保存】按钮，即可将前台模板另存为后台模板。

在后台模板中，首先将页面上方的图片文件 Top.gif 改为文件 Admin_Top.gif，然后将图片下的文本“后台管理”改为“返回网络相册主页”，并将其链接更改为“../Default.aspx”。同时，将嵌套的表格 Table2 的第 1 单元格的宽度设置为 130 像素，第 2 个单元格的宽度设置为 650 像素。最后，删除页面左边的可编辑区域 EditRegion3，并将可编辑区域 EditRegion4 的起始标记置于页面中的嵌套表格 Table2 的起始标记之前，将可编辑区域 EditRegion4 的结束标记置于表格 Table2 的结束标记之后。也就是说，整个表格 Table2 将置于可编辑区域 EditRegion4 之内，如图 9.48 所示。

页面的左侧将显示后台的功能菜单，而右侧则将显示相应的功能操作界面。

下面，将在模板页面的左侧添加后台的功能菜单。这里，将使用 ASP.NET 2.0 中新增的 TreeView 控件。在 ASP.NET 1.x 中，微软并未将其作为 ASP.NET 的一个内置服务器控件，而是将其作为一个附加的 Web 控件集(即 Internet Explorer Web)中的一部分。也就是说，在 ASP.NET 1.x 中，如果需要使用 TreeView 控件，用户则需先下载并安装 IE Web 控件，然后将其程序集复制到 Bin 目录下，方可进行操作。但对 ASP.NET 2.0 来说，则不存在此问题，它已作为一个内置的服务器控件存在。

遗憾的是，目前 Dreamweaver 8 并不支持对 ASP.NET 2.0 的可视化操作。也就是说，我们无法在 Dreamweaver 8 操作界面上的【ASP.NET-插入】工具栏中可视化地拖动并插入 ASP.NET 2.0 中新增的控件。这也是 Dreamweaver 8 针对 ASP.NET 的一个重大缺陷，但相信，Dreamweaver 的新版本将会全方位地提供对 ASP.NET 2.0 的支持。

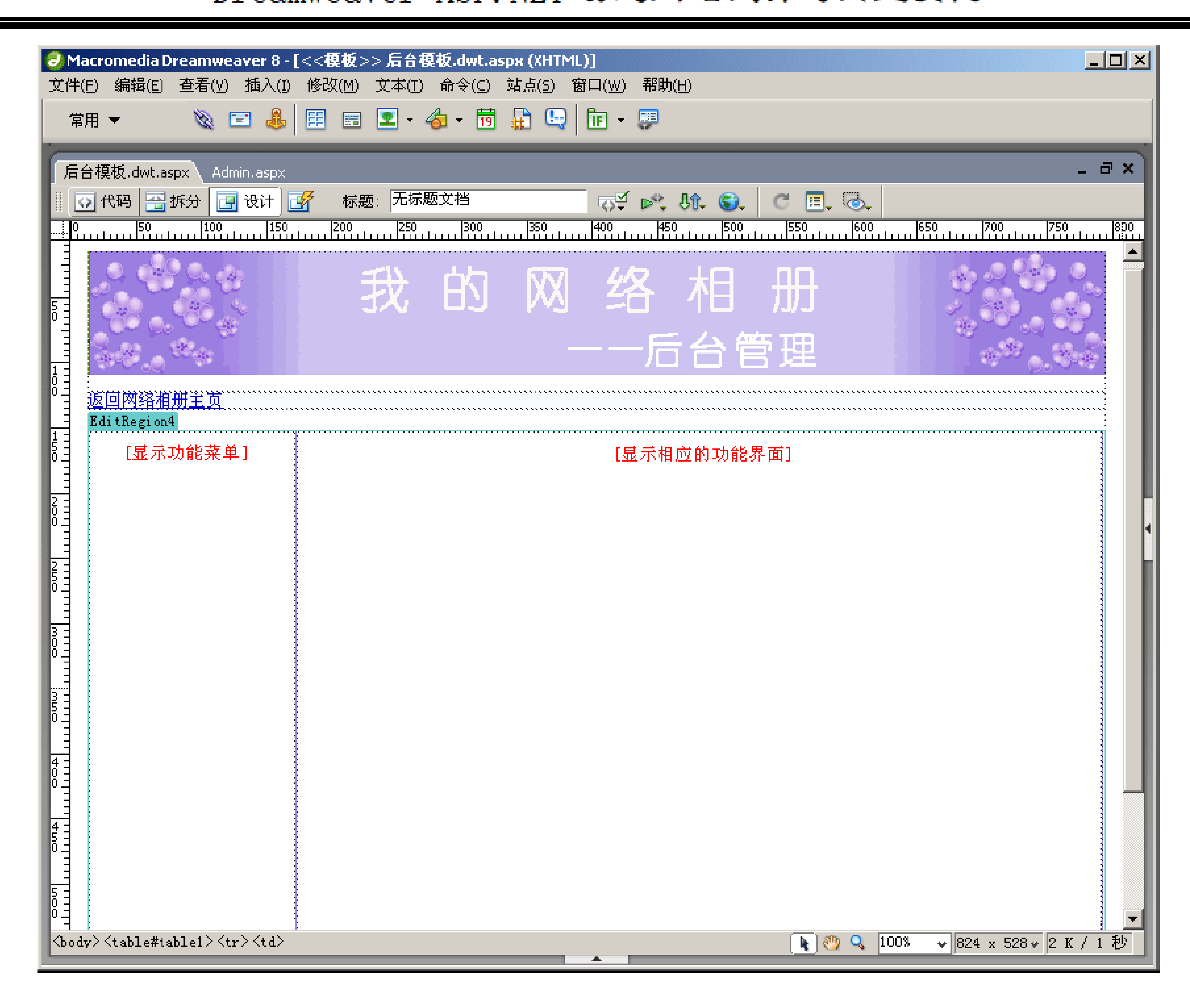


图 9.48　后台模板的结构

但是，不要忘记 Dreamweaver 从本质上来说仍是一个代码编辑器，我们完全可以在其【代码】视图中编写代码，而无须在【设计】视图中进行操作，这对在 Dreamweaver 8 中处理 ASP.NET 2.0 中新增的服务器控件无疑是一个最简单，也最直接的方法。只是，这个方法要求我们对 ASP.NET 2.0 中新增控件的语法非常清楚。

接下来，切换至 Dreamweaver【代码】视图，在可编辑区域 EditRegion4 的起始标记与表格 Table2 的起始标记之间添加表单的起始标记<Form Id="form1" Runat="Server">，然后在可编辑区域 EditRegion4 的结束标记与表格 Table2 的结束标记之间添加表单的结束标记</Form>。也就是说，将整个表格 Table2 置于表单 Form1 中，而将表单 Form1 置于可编辑区域 EditRegion4 之内。

然后，在表格 Table2 的第 1 个单元格中输入功能菜单的相关代码，其代码如下：

【示例代码】

```
<asp:TreeView ID="TreeView2" runat="server" ExpandDepth="1"
ShowLines="True" ForeColor="Black">
  <Nodes>
    <!—注释：添加父结点“分类管理”，并设置其选择操作为展开-->
    <asp:TreeNode SelectAction="Expand" Text="分类管理">
      <!—注释：添加子结点“添加分类”，设置链接为“Class_Add.aspx”-->
      <asp:TreeNode NavigateUrl="Class_Add.aspx" Target="_self" Text="添加
分类"></asp:TreeNode>
      <!—注释：添加子结点“管理分类”，设置链接为“Class_Admin.aspx”-->
      <asp:TreeNode NavigateUrl="Class_Admin.aspx" Target="_self" Text="管
理分类"></asp:TreeNode>
    </asp:TreeNode>
```

```
    <!—注释：添加父结点“相册管理”，并设置其选择操作为展开-->
    <asp:TreeNode SelectAction="Expand" Text="相册管理">
      <!—注释：添加子结点“上传相片”，设置链接为“Photo_Add.aspx”-->
      <asp:TreeNode NavigateUrl="Photo_Add.aspx" Target="_self" Text="上传
相片"></asp:TreeNode>
      <!—注释：添加子结点“管理相片”，设置链接为“Photo_Add.aspx”-->
      <asp:TreeNode NavigateUrl="Photo_Admin.aspx" Target="_self" Text="管
理相片"></asp:TreeNode>
    </asp:TreeNode>
  </Nodes>
</asp:TreeView>
```

注意，以上注释仅为方便读者理解代码而添加的。在实际的代码中，是不允许在 TreeView 控件的定义代码内添加注释的。

切换至【设计】视图，可以发现在表格 table2 的第 1 个单元格中仅显示了标记，而没有正常显示出所定义的功能菜单，这就是 Dreamweaver 不支持对 ASP.NET 2.0 的可视化操作所导致的。但是，在浏览器中访问此页面，可以发现功能菜单能够正常显示，如图 9.49 所示。

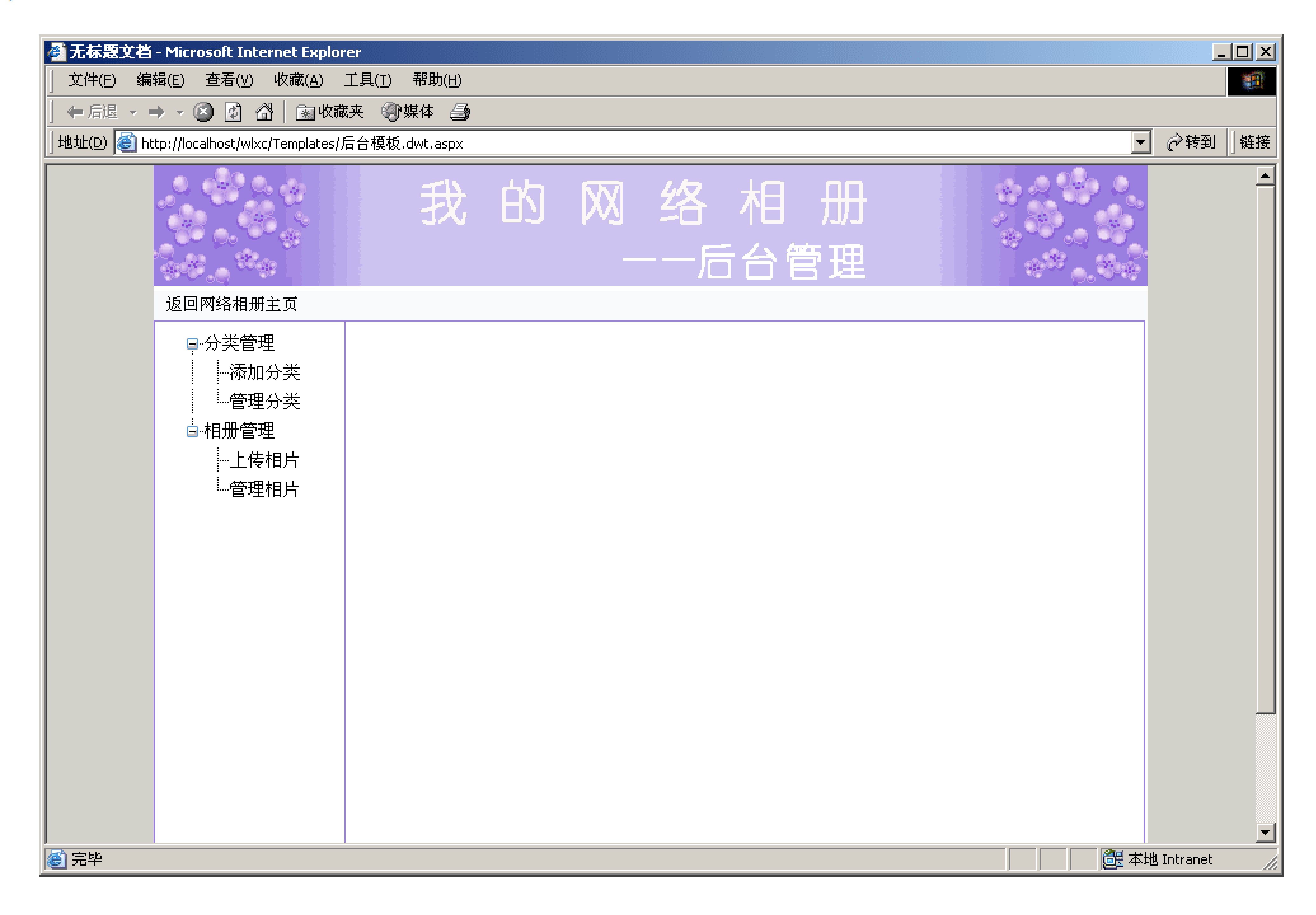


图 9.49　预览后台模板

至此，后台模板创建完成。最后，有些读者也许会对模板的设计提出疑问，为什么可编辑区域 EditRegion4 要包括整个表格 table2，而不是 table2 的第 2 个单元格，按理说功能菜单也应属于不可编辑的区域呀？以下几点也许能帮助大家解除疑惑。

- 当使用【服务器行为】面板中的【插入记录】和【更新记录】命令时，将会对表单 Form 标记进行编辑，这意味着表单只能存在于可编辑区域中，否则在后面的功能页面中将可能无法使用模板。

- TreeView 控件必须处于表单 Form 标签之内，且该 Form 必须带有“Runat="Server"”属性。
- 整个页面只能有一个 Form 标签。

以上几点决定了可编辑区域 EditRegion4 只能在表单 Form 之外，而表单 Form 又必须包含整个表格 Table2。这就是，功能菜单必须处于可编辑区域之内的原因，尽管在调用该模板的后台功能页面中并不会对其进行编辑。

9.3.2　后台首页

后台的首页相当简单，其结构与模板页类似。它并不提供任何的功能操作，而仅仅是起一个中间过渡的作用，其对应的文件名为 Admin.aspx。

新建一个 ASP.NET VB 类型的页面，将其命名为 Admin.aspx。选择【修改】|【模板】|【套用模板到页】命令，在弹出的【选择模板】对话框中，选择前面创建的【后台模板】，如图 9.50 所示。

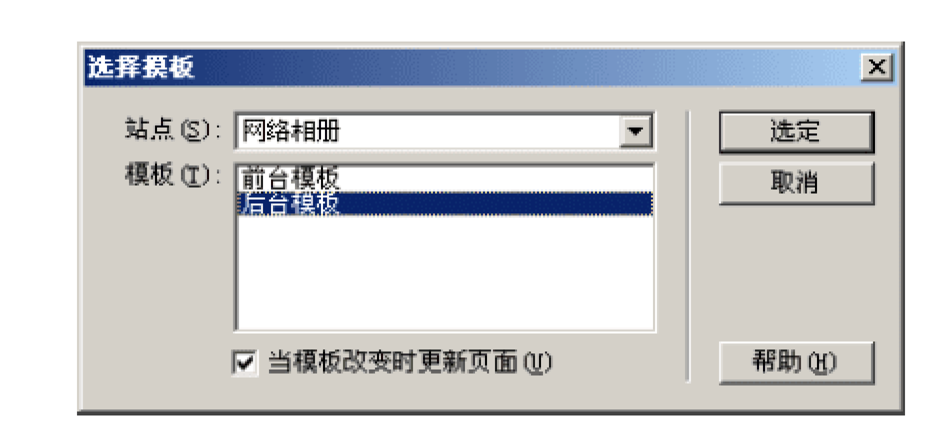


图 9.50　【选择模板】对话框

单击【选定】按钮，将模板应用到本页面。

修改页面标题为“后台管理”，将光标移至表格 table2 的第 2 个单元格中，添加文本“请选择您所要执行的操作”，并设置其水平对齐方式为“居中对齐”，垂直对齐方式为“顶端”。

至此，后台首页制作完成，页面预览如图 9.51 所示。

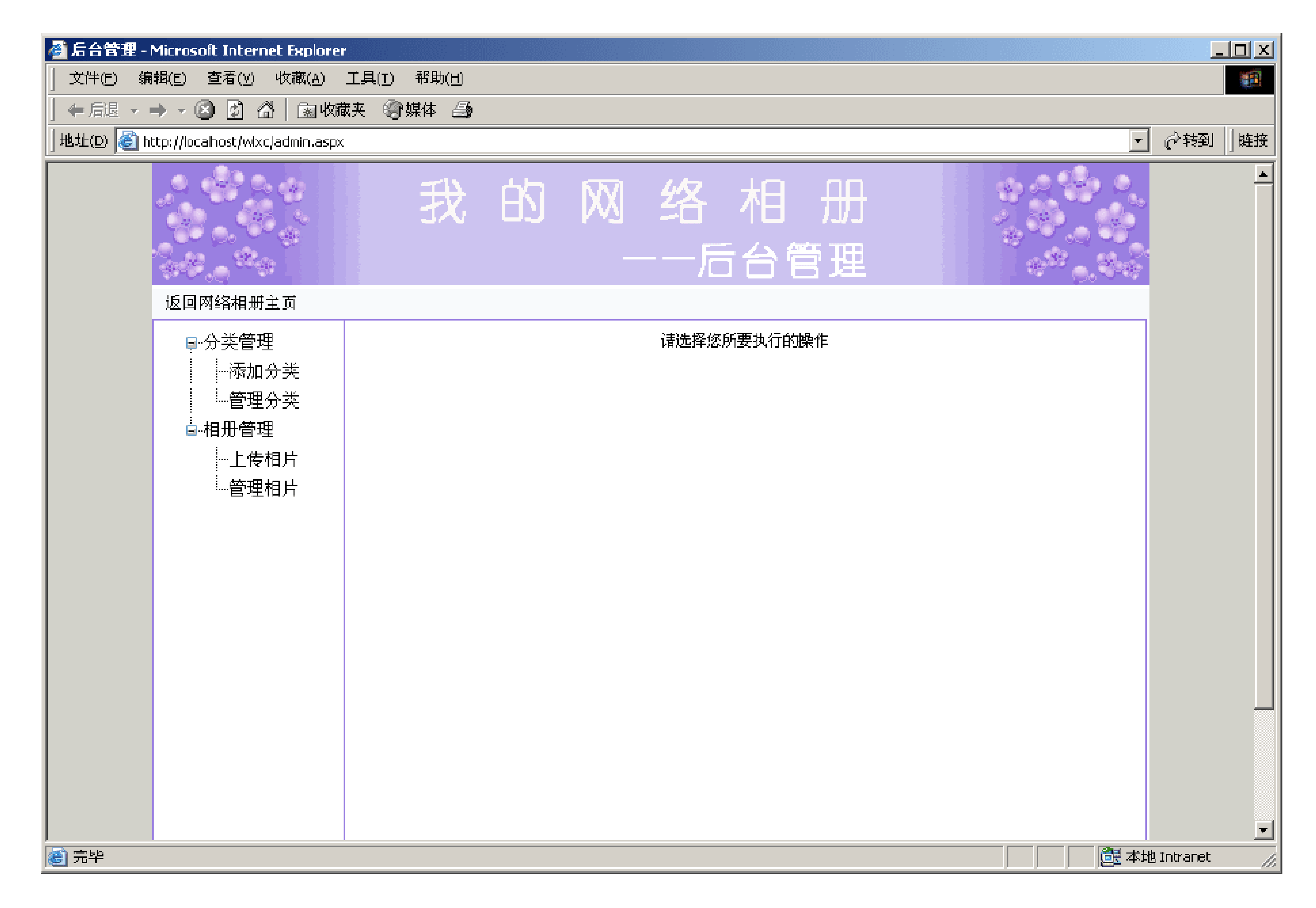


图 9.51　页面预览

9.3.3　添加分类

添加分类主要用于设置相片的分类信息，其所需设置的信息仅包括分类名称。

新建一个 ASP.NET VB 类型的页面，将其命名为 Class_Add.aspx。选择【修改】|【模板】|【套用模板到页】命令，在弹出的【选择模板】对话框中，选择【后台模板】，将模板应用到本页面。

修改页面标题为“添加分类”，然后在表格 table2 的第 2 个单元格中，添加功能标题“添加分类信息”和文本“分类名称：”，同时添加一个文本框输入控件 Name 和一个提交按钮控件 Add，如图 9.52 所示。

图 9.52　页面设计

为了保证分类名称的输入不为空，还须在【添加】按钮下插入一个 RequiredFieldValidator 验证控件，其属性设置如图 9.53 所示。

图 9.53　插入 RequiredFieldValidator 控件

打开【应用程序】面板组，选择【服务器行为】面板，单击 + 按钮，从弹出的菜单中选择【插入记录】命令，此时将弹出【插入记录】对话框，如图 9.54 所示。

在【提交值，自】选项中选择 form[0]，在【连接】选项中选择 Cnn，在【插入到表格】

选项中选择数据表 ClassInfo。然后，在【列】列表框中，设置字段 ClassName 的获取值为表单元素 name。最后，设置提交成功后所转向的页面为本页面，即 Class_Add.aspx。单击【确定】按钮，完成插入记录的设置。

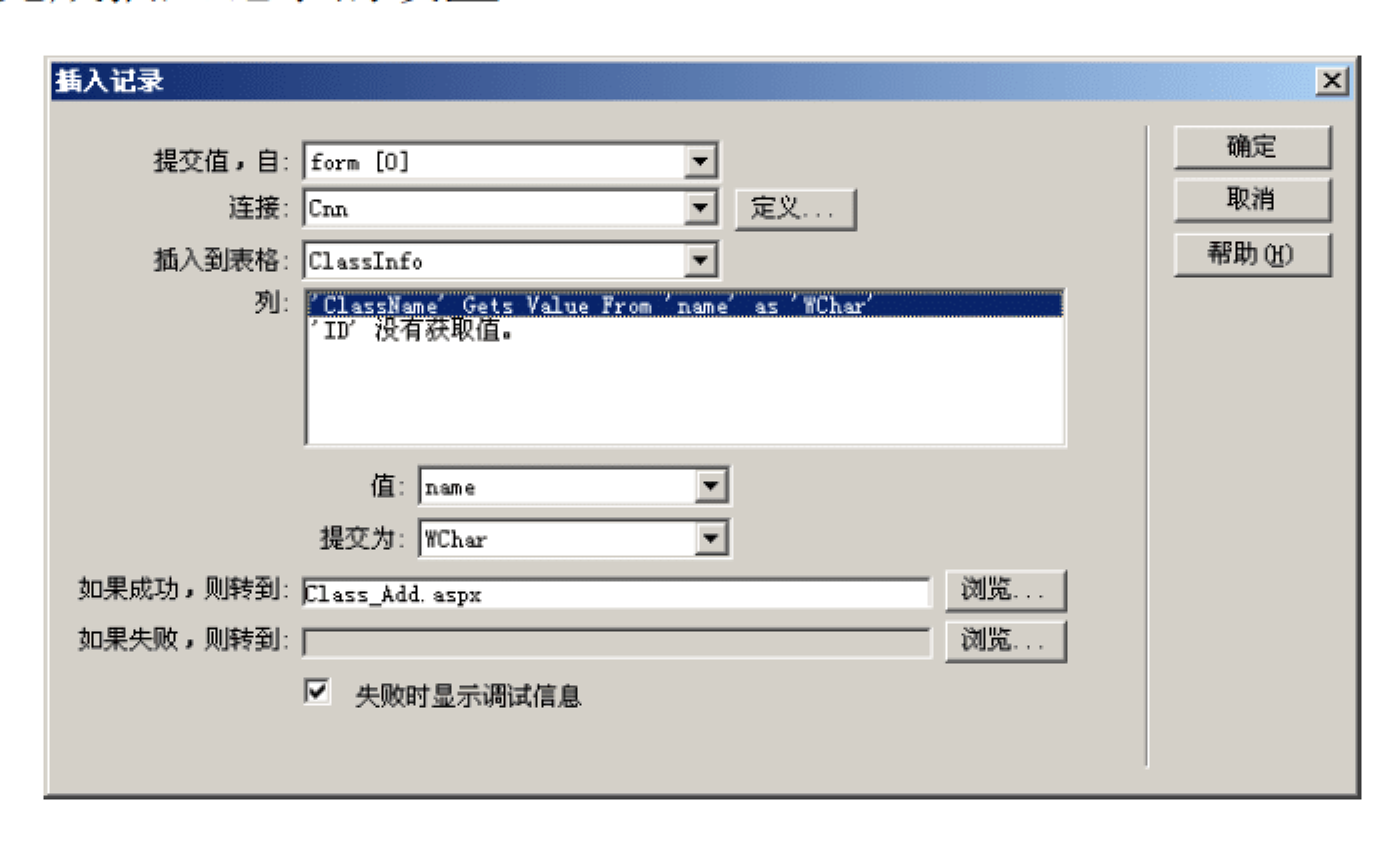


图 9.54　插入记录

至此，添加分类信息的页面制作完成，页面预览如图 9.55 所示。

图 9.55　页面预览

9.3.4　管理分类

管理分类主要用于对相片的分类信息进行管理，其功能包括查询相片的分类信息、编辑或删除指定的相片分类信息。

(1) 新建一个 ASP.NET VB 类型的页面，将其命名为 Class_Admin.aspx。选择【修改】|【模板】|【套用模板到页】命令，在弹出的【选择模板】对话框中选择【后台模板】，将模板应用到本页面。

修改页面标题为“分类管理”，然后在表格 table2 的第 2 个单元格中，添加功能标题

“分类管理”。

接下来将通过数据网格(即 ASP.NET 中的 DataGrid 数据控件)来显示相片的分类信息。在此之前，需要创建一个数据集。

(2) 在【应用程序】面板组中，切换至【服务器行为】面板，单击按钮，在弹出的菜单中选择【数据集】命令。在弹出的【数据集】对话框中，设置数据集名称为 DataSet1，连接选择 Cnn，表格选择 ClassInfo，筛选条件为“无”，排序方式选择按字段 ID 进行升序排序，如图 9.56 所示。

图 9.56　创建数据集 DataSet1

单击【确定】按钮，完成数据集的创建。

(3) 将光标置于功能标题之下，在【服务器行为】面板中，单击按钮，在弹出的菜单中选择【数据网格】命令，此时将弹出【数据网格】对话框，如图 9.57 所示。

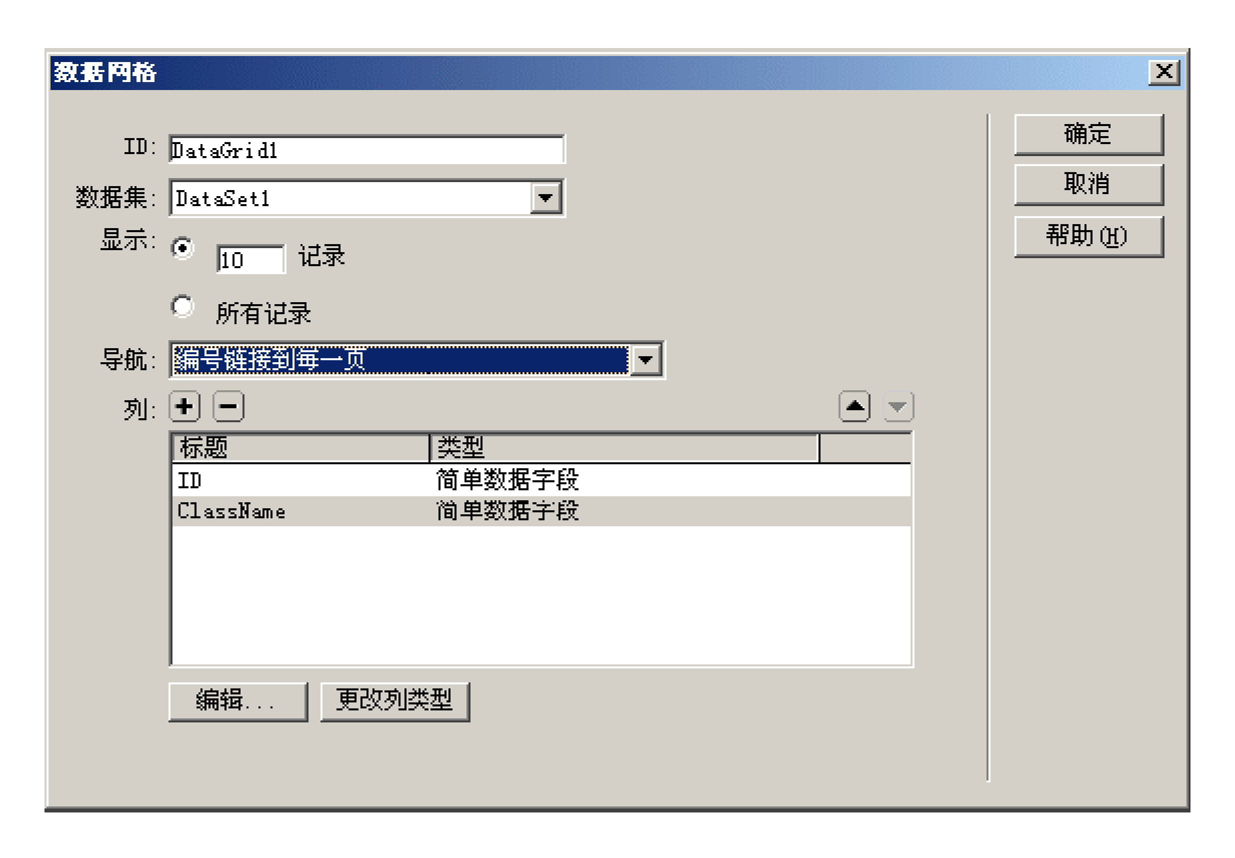


图 9.57　数据网格

设置 ID 为 DataGrid1，数据集选择 DataSet1，设置一次显示 10 条记录，在【导航】下拉列表框中选择【编号链接到每一页】。单击【添加列】按钮，在弹出的菜单中选择【'编辑'、'更新'、'取消'按钮】命令，如图 9.58 所示。

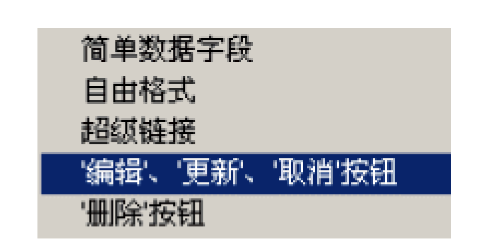


图 9.58　关联菜单

在弹出的【编辑、更新、取消按钮列】对话框中，设置【标题】为“编辑”，【按钮类型】为“链接按钮”，【更新表格】

为 ClassInfo，【主键】为 ID，【提交为】为 Integer，如图 9.59 所示。

图 9.59　【编辑、更新、取消按钮列】对话框

单击【确定】按钮，完成编辑列的添加。

(4) 再次单击【添加列】按钮，在弹出的菜单中选择【'删除'按钮】，在弹出的【删除按钮列】对话框中设置【标题】为“删除”，【按钮类型】选择“链接按钮”，【删除自】为 ClassInfo，【主键】为 ID，【提交为】为 Integer，如图 9.60 所示。

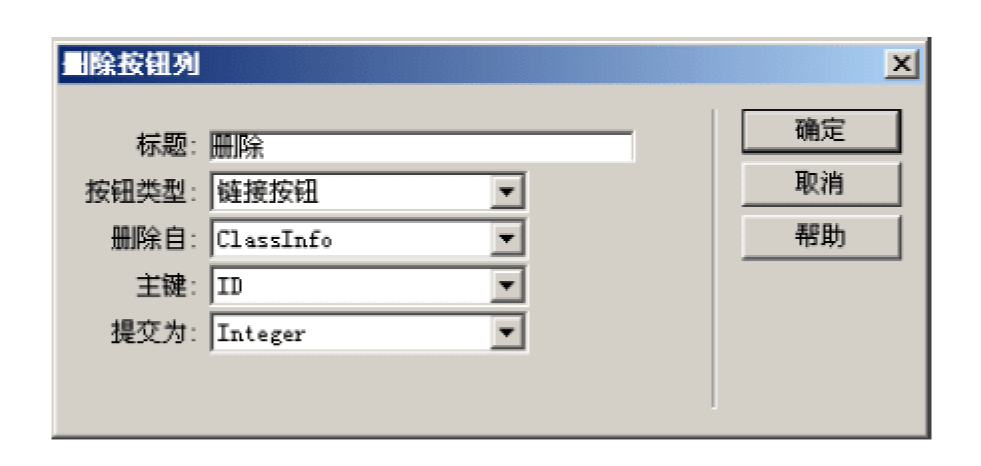


图 9.60　【删除按钮列】对话框

单击【确定】按钮，完成删除列的添加。

在【列】列表框中，选择 ID 列，单击【编辑】按钮，在弹出的【简单数据字段列】对话框中，设置该列的显示标题为“分类编号”，如图 9.61 所示。

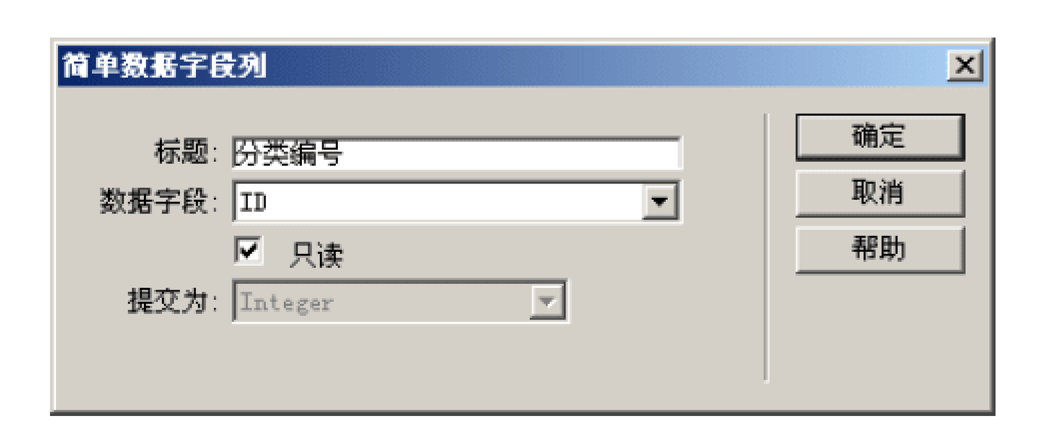


图 9.61　设置 ID 列的属性

(5) 然后再选择 ClassName 列，单击【编辑】按钮，在弹出的【简单数据字段列】对话框中，设置该列的显示标题为“分类名称”，同时取消【只读】复选框的选择，如图 9.62 所示。

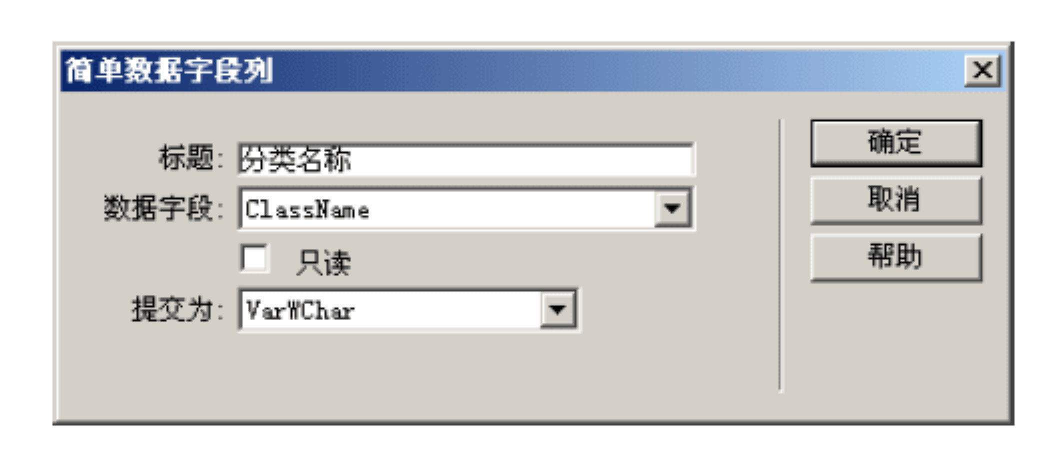


图 9.62　设置 ClassName 列的属性

此时，在数据网格中所需显示的各项列均设置完成，如图 9.63 所示。

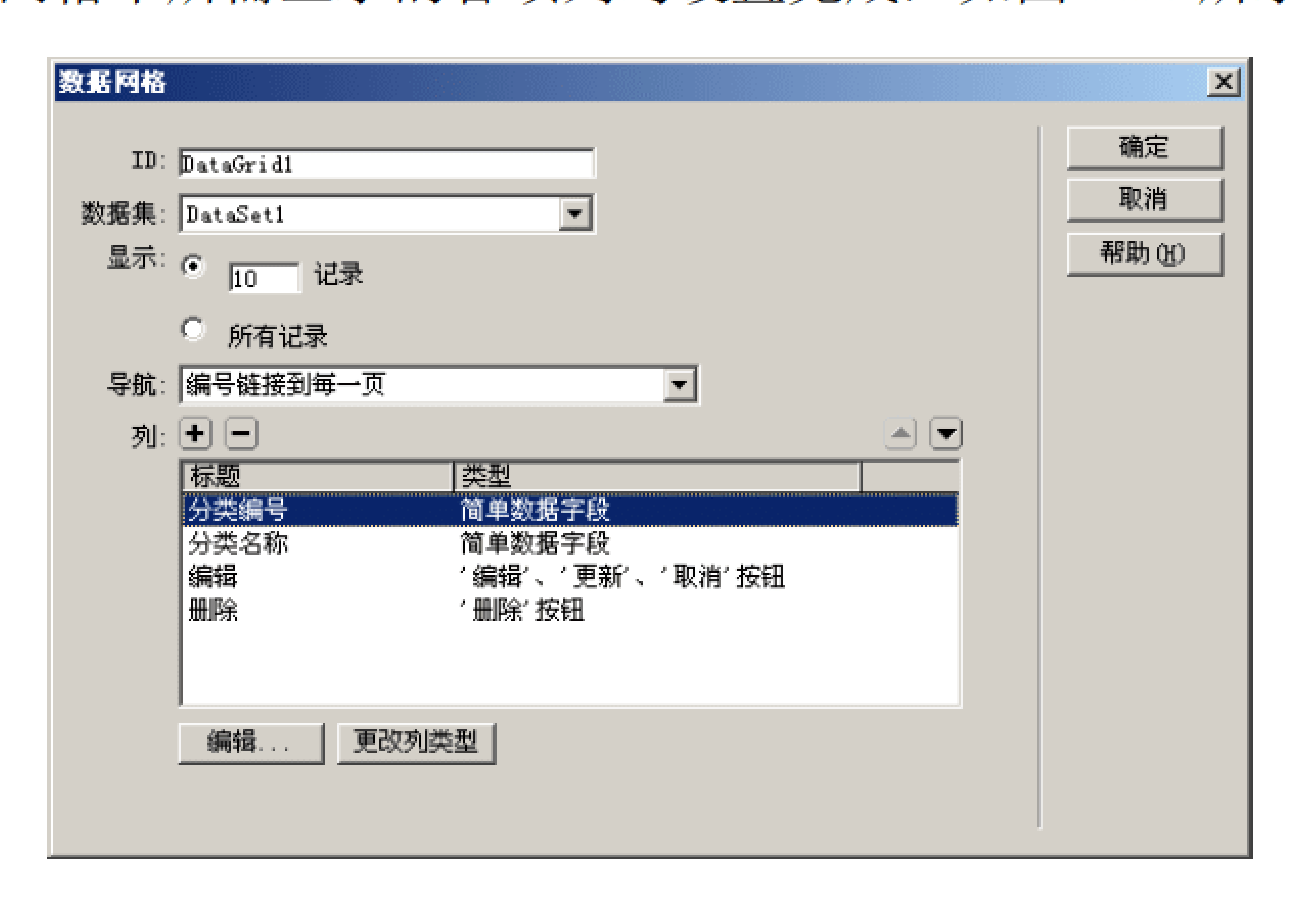


图 9.63 数据网格中列的设置

(6) 单击【确定】按钮，完成数据网格的创建。这时，还需对数据网格的显示作进一步调整。在【设计】视图中，选择所创建的数据网格并右击，从弹出的快捷菜单中选择【编辑标签】命令，在弹出的【标签编辑器】对话框中，选择【布局】项，设置【宽度】为 400 像素，【单元格边距】为 3，【单元格间距】为 0，【边框宽度】为 1，如图 9.64 所示。

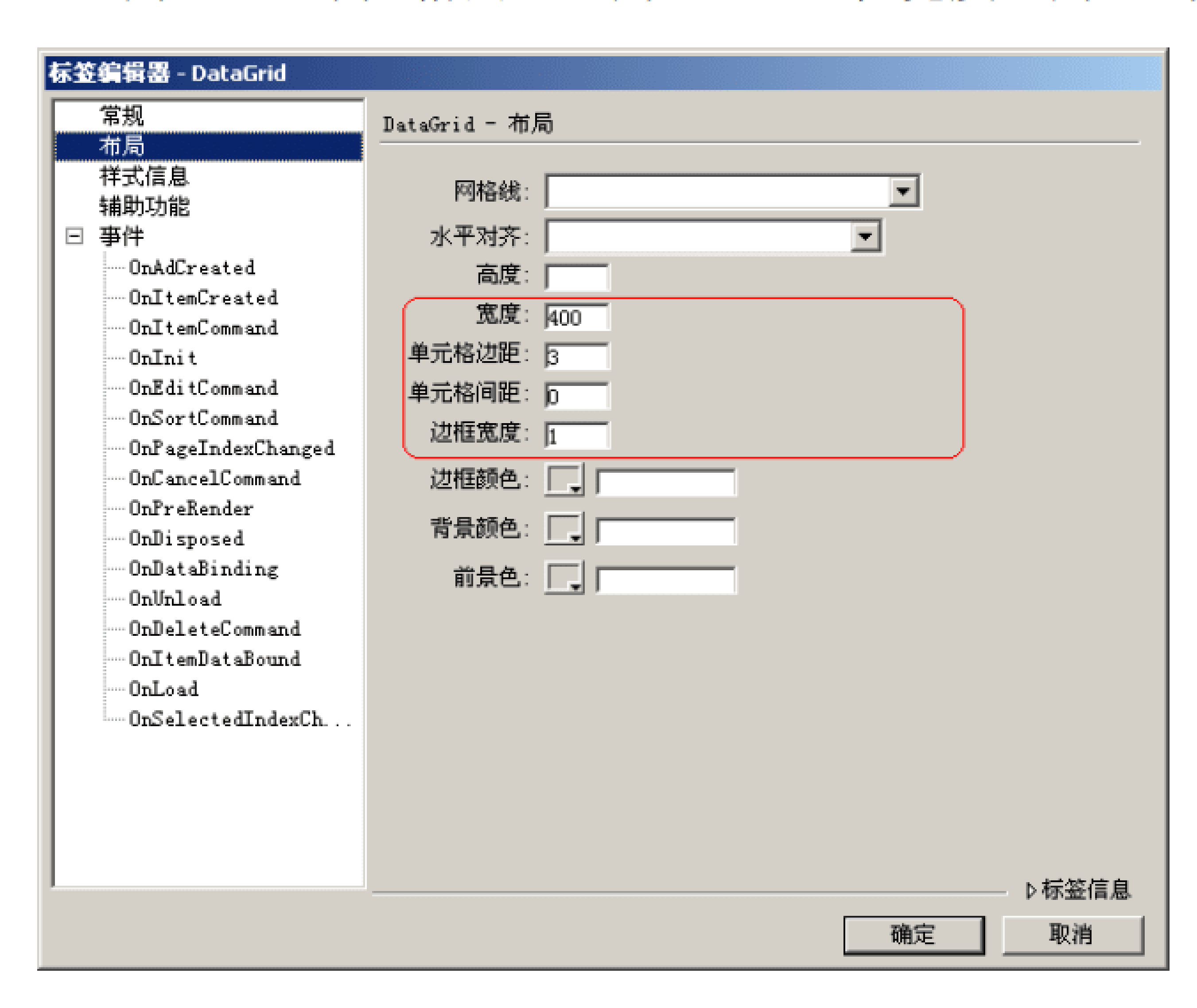


图 9.64 【标签编辑器】对话框

单击【确定】按钮，完成对数据网格布局的设置。

至此，页面的功能设计完成，页面预览如图 9.65 所示。

(7) 单击相应分类的【编辑】按钮，其分类名称将显示为一个文本框，同时【编辑】按钮也变为【更新】按钮和【取消】按钮，如图 9.66 所示。

在文本输入框中修改分类名称，单击【更新】按钮，即可更新所修改的记录，并返回初始状态；单击【取消】按钮，可取消更新操作。此外，单击【删除】按钮，可删除指定的分类信息。

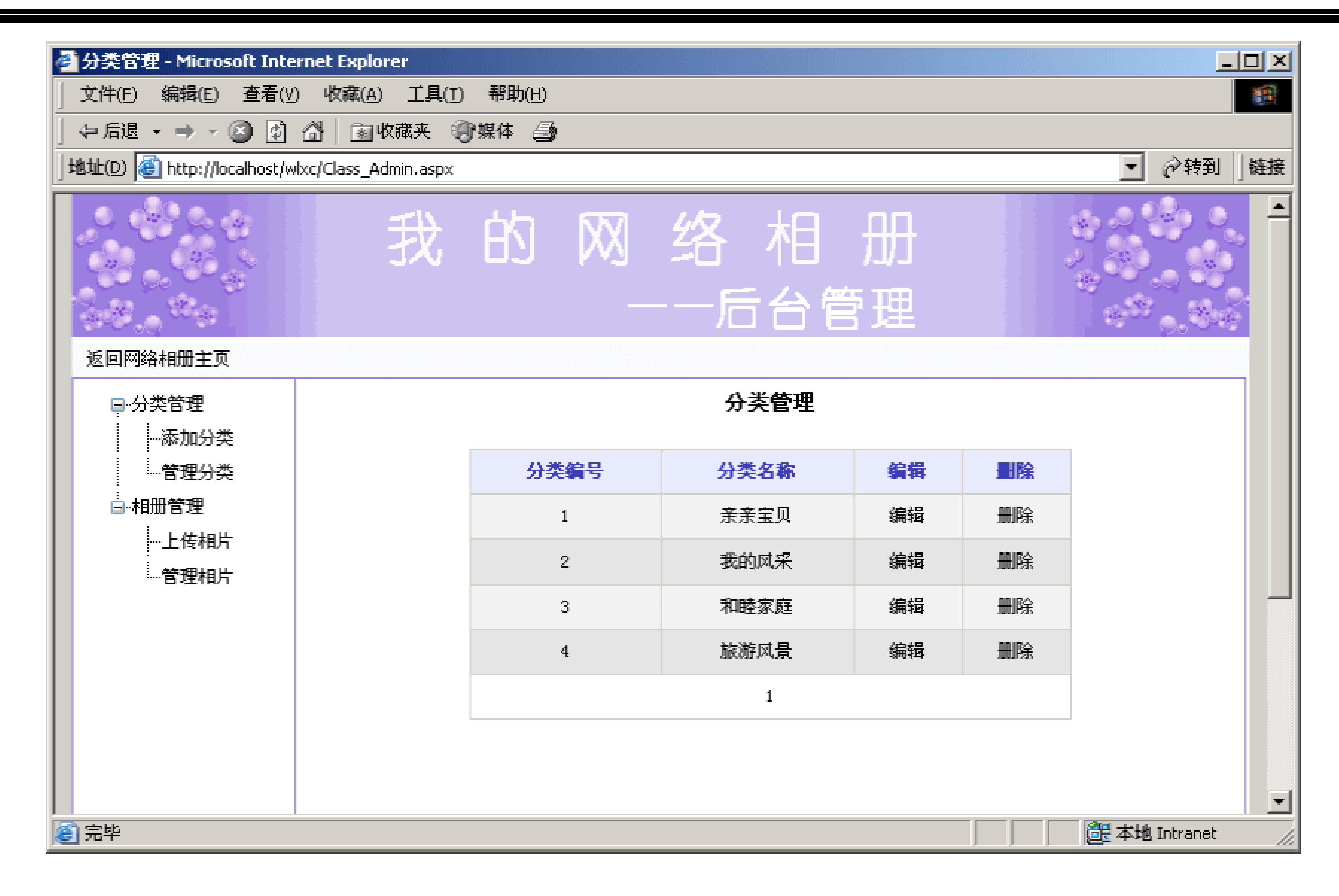


图 9.65　页面预览

图 9.66　编辑状态

需要说明的是，必须将页面绑定的 PostBackBind 属性设置为 False(如图 9.67 所示)；否则，更新操作无法正常执行。

页面功能已经基本完成，但是细心的读者也许会发现对于分类信息的删除功能仍有缺陷：当所要删除的分类下存在相片信息时，该分类应该不允许被删除，而这里并没有进行判断。确实，这里删除调用的是 Dreamweaver 8 中默认的数据集 DataSet1 的 OnDataGridDelete 事件，因此不会对特殊情况进行处理。但用户可以通过自定义的事件代码来实现。

(8) 在【设计】视图中，选择所创建的数据网格并右击，从弹出的快捷菜单中选择【编辑标签】命令，在弹出的【标签编辑器】对话框中，选择【事件】|OnItemCommand 分类，

如图 9.68 所示。

```
  <EditOps>
    <EditOpsTable Name="ClassInfo" />
    <Parameter Name="ClassName" Type="VarWChar" />
    <Parameter Name="ID" Type="Integer" IsPrimary="true" />
  </EditOps></MM:DataSet>
<MM:PageBind runat="server" PostBackBind="False" />
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
```

图 9.67　修改代码

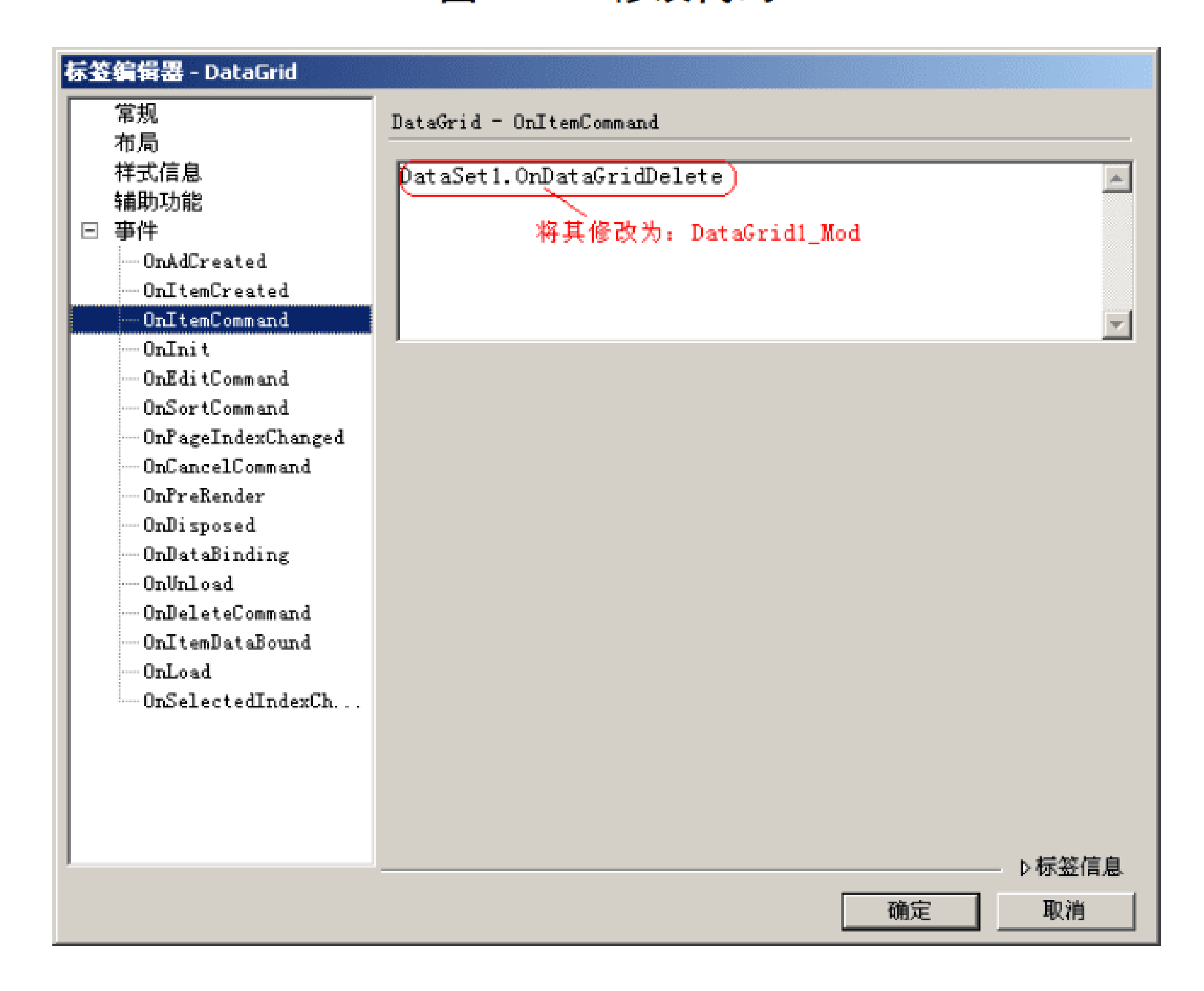


图 9.68　OnItemCommand 事件

从图 9.68 所示的右边的列表中可以看到，DataGrid 控件默认的 OnItemCommand 事件为"DataSet1. OnDataGridDelete"，将其修改为自定义的事件 Datagrid1_Mod。单击【确定】按钮，关闭对话框。

(9)　然后，切换至【代码】视图，添加事件代码，如下所示：

【示例代码】

```
<script language="vb" runat="server">
  Sub DataGrid1_Mod(ByVal Sender As Object, ByVal E As
DataGridCommandEventArgs)
    Dim codestr As String = E.Item.Cells(0).Text  '获取所要删除的分类 ID
    Dim DataR As OleDbDataReader
    Dim Cnn As OleDbConnection
    Dim Cmd As OleDbCommand
    Dim StrCnn As String
    Dim Sql As String
    If CType(e.CommandSource, LinkButton).CommandName = "Delete" Then
      '获取数据库连接字符串
      StrCnn = System.Configuration.ConfigurationSettings.AppSettings
("MM_CONNECTION_STRING_Cnn")
      '创建数据库连接
      Cnn = New OleDbConnection(StrCnn)
```

```
        Cnn.Open()
        '查询数据库中是否存在此分类的相片信息
        Sql = "select * from photoinfo where classid=" & Trim(codestr)
        Cmd = New OleDbCommand(Sql, Cnn)
        DataR = Cmd.ExecuteReader()
        If DataR.Read() Then
          '存在此分类下的相片信息，提示无法删除
          ClientScript.RegisterStartupScript(Me.GetType(), "", GetInfo("该分
类下存在相片信息，无法删除！"))
          Cnn.Close()    '关闭数据库连接
        Else
          '不存在该分类下的相片信息
          DataR.Close()  '关闭所打开的 OleDbDatareader 对象
          '执行删除操作
          Sql = "delete from classinfo where id=" & codestr
          Cmd = New OleDbCommand(Sql, Cnn)
          Cmd.ExecuteNonQuery()
          Cnn.Close()
          '将页面重新跳转至本页面，更新数据的显示
          Response.Redirect("Class_Admin.aspx")
        End If
      End If
    End Sub
    '定义函数 GetInfo，用于获取格式化后的错误提示信息
    Function GetInfo(ByVal Str1 As String) As String
      Dim Str2 As String
      Str2 = "<script language='javascript'>alert('" & str1 & "')<"
      Str2 += "/"
      Str2 += "script>"
      Return Trim(Str2)
    End Function
</script>
```

以上代码的功能是，首先判断数据库中是否存在指定分类下的相片信息。如果存在，则调用 ClientScript.RegisterStartupScript()方法提示相应的错误信息，其中所提示的信息通过自定义函数 GetInfo()来返回格式化脚本后的信息；如果不存在指定分类下的相片信息，则直接将其删除，并重定向本页面，更新数据的显示。

提示：在此页面中，必须导入命名空间 System.Data 和 System.Data.OleDb。

(10) 重新预览本页面，当删除存在相关相片的分类时，系统将会弹出错误提示，取消删除操作，如图 9.69 所示。

图 9.69　提示信息

至此，页面功能全部完成。从这里也可以看出，虽然 Dreamweaver 8 提供了非常快捷的方式来对数据进行操作，但其数据库行为只能提供相对简单的功能(尽管这对于一些基本的数据操作已经足够了)。如果需要在其中加入一些特殊的判断或功能，则仍需通过代码来实现，这就要求我们对 ASP.NET 中的数据控件的语法有足够的了解。所以，请读者不要忘记，Dreamweaver 只是一种辅助的 ASP.NET 开发工具，掌握 ASP.NET 的各种语法才是最根本的。

9.3.5　上传相片

现在介绍上传相片功能的实现。在此功能中，除了上传指定的相片外，还应在上传相片之前提供相片预览的功能，这样可以避免上传错误的相片。

(1)　新建一个 ASP.NET VB 类型的页面，将其命名为 Photo_Add.aspx。选择【修改】|【模板】|【套用模板到页】命令，在弹出的【选择模板】对话框中，选择【后台模板】，将模板应用到本页面。

(2)　修改页面标题为“上传相片”，然后在表格 table2 的第 2 个单元格中，添加功能标题“上传相片”。在功能标题下方插入一个 4 行 4 列的表格，在其中添加相应的文本和控件，并在表格下方添加一个 Image 服务器控件，如图 9.70 所示。

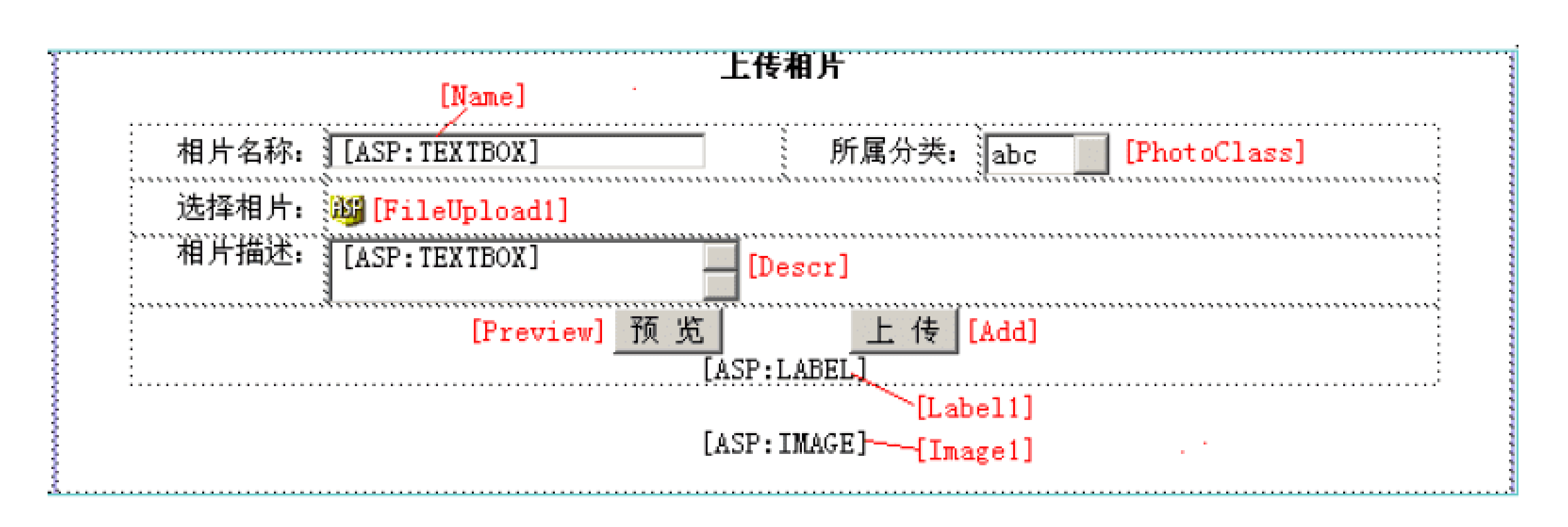


图 9.70　界面设计

其中，ID 名为 FileUpload1 的控件为 ASP.NET 2.0 中新增的 FileUpload 服务器控件，该控件包含一个文本框和一个【浏览】按钮，它主要用于实现文件的上传。由于 Dreamweaver 8 对 ASP.NET 2.0 可视化操作的不支持，因此这里仅显示为标记。该控件无法在【设计】视图中直接插入，而只能在【代码】视图中编写代码，其定义代码如图 9.71 所示。

```
            <TR>
              <TD height="25" align=right style="padding-right:5px;padding-top:5px;
padding-bottom:5px;">选择相片：</TD>
              <TD style="padding-left:5px" align="left" colspan=3>
                  <asp:FileUpload ID="FileUpload1" runat="server" ToolTip="选择相片"
Width="400px" EnableTheming="True" /></TD>          定义FileUpload服务器控件
            </TR>
            <TR>
```

图 9.71　定义 FileUpload 控件

在表格的第 4 行添加了一个【预览】按钮。其作用是，在上传文件之前获取文本框中的文件路径，并将其赋予表格下方的 Image 控件的 ImageUrl 属性，从而达到预览相片的目的。

但是，这里将会出现两个问题。

①　FileUpload 控件并没有提供相关属性可以获取文本框中的文件全路径(注意，是文件全路径，而不是文件名)。

② 经过测试发现，作为 FileUpload 服务器控件的一部分，其文本框的状态没有通过回发进行保存。也就是说，当数据回发至服务器时，其文本框的值将会丢失，这也是出于一种安全风险的考虑。在这种情况下，如果【预览】按钮为 ASP.NET 中的服务器控件，则单击时势必会导至数据回发。即使此时能够预览相片，其文件输入框中的值也会清空，用户又得重新选择，这会给用户的操作带来一定的麻烦。

所幸的是，在页面预览后，通过查看源文件可以发现 FileUpload 服务器控件在服务器执行后将转化为 HTML 中的<Input type="File">标签，而 Image 服务器控件也将转化为 HTML 中的<Img>标签。既然如此，那完全可以通过 JavaScript 脚本来获取文本框的值并将其赋给<Img>标签的 src 属性，从而达到预览的目的。

因此，这里所添加的【预览】按钮并非 ASP.NET 中的 Button 服务器控件，而是 HTML 中的<Input type="button">标签，如图 9.72 所示。

```
            <TD height="35" colspan="4" align=center>
            <input type="button" name="preview" value="预 览" onClick=
"javascript:dispphoto();">                    预览按钮
                   
             <asp:Button id="Add" runat="server"  onclick="Save_Click" Width=
"50px" Text="上 传"></asp:Button>
             <br />
```

图 9.72 【预览】按钮的定义代码

在【预览】按钮的定义中，设置了其单击事件为 JavaScript 脚本函数 dispphoto()，该函数的定义如图 9.73 所示。

```
<title>上传相片</title>
<script language="javascript">
function dispphoto(){
  if (document.getElementById("FileUpload1").value!="")
  {
   document.getElementById("Image1").style.width="200px";  设置Image控件的图像宽度
   document.getElementById("Image1").src=document.getElementById("FileUpload1").value;
  }                      获取文件输入框的值并将其赋予Image控件的src属性
}
</script>
```

图 9.73 定义 dispphoto 函数

这样，通过 JavaScript 与 ASP.NET 的结合，轻易地实现了预览相片的功能。这里还需说明的是，当用户初次加载页面时，Image 服务器控件的 ImageUrl 属性为空，不显示任何图像，为了达到隐藏的目的，在定义时特意将其 Width 属性设置为 0。因此，这里在预览相片时，必须在 dispphoto 脚本函数中设置 Image 的宽度，否则图像是无法显示的。

下面，再来看看相片所属分类对应的下拉列表控件 PhotoClass 的添加。该控件主要用于显示相片的分类信息，以便用户进行选择。在插入该控件之前，需要创建一个数据集，以便绑定数据的显示。

(3) 打开【应用程序】面板组，选择【服务器行为】面板，单击 + 按钮，在弹出的菜单中选择【数据集】命令。在弹出的【数据集】对话框中，选择表格 ClassInfo，并设置排序方式为按字段 ID 升序排序，如图 9.74 所示。

单击【确定】按钮，完成数据集的创建。

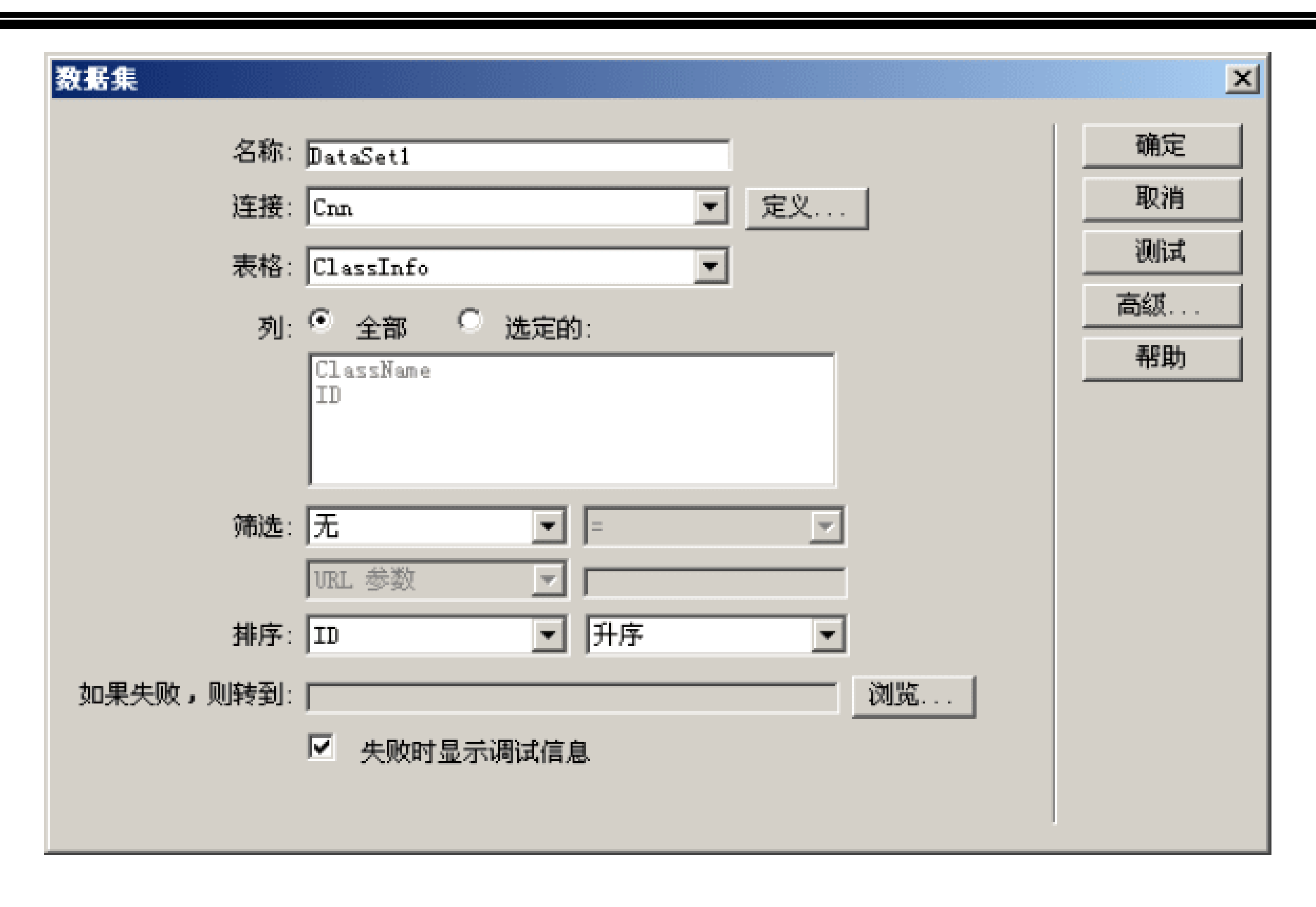


图 9.74　创建数据集 DataSet1

(4) 在插入 DropDownList 控件时，在弹出的【asp:下拉列表】对话框中，需对其进行数据绑定，如图 9.75 所示。

图 9.75　创建下拉列表控件

这里，数据集选择前面所创建的数据集 DataSet1，获取标签选择 DataSet1 中的字段 ClassName，而获取值则选择字段 ID。

至此，页面的界面设计及数据绑定基本完成。最后，再来看【上传】按钮，其操作可分为两步：首先，上传选定的相片文件至指定的文件夹中；然后，将相片信息保存至数据库中。由于其操作不仅仅是向数据库中插入数据，因此这里数据的提交不能使用【服务器行为】面板中的【插入记录】命令，而只能通过代码来实现。

(5) 为此，定义【上传】按钮的单击事件为 Save_Click，其事件代码如下：

【示例代码】

```
Sub Save_Click(ByVal Sender As Object, ByVal E As EventArgs)
  Dim Sql As String
  Dim Tag As Integer = 0  '定义变量 Tag，用于标识上传文件是否成功
  Dim NameStr As String = name.Text  '获取相片名称信息
  Dim DescrStr As String = Descr.Text  '获取相片描述信息
  Dim theClass As String = photoclass.SelectedItem.Value  '获取相片所属分类
  Dim PathStr As String
  Dim ErrStr As String
  Dim Cnn As OleDbConnection
  Dim Cmd As OleDbCommand
  Dim StrCnn As String
  '调用 FileUpload 控件的 HasFile 方法判断用户是否选择了相片文件
```

```
      If FileUpload1.HasFile Then
        '用户选择了文件，开始上传
        Try
          '设置文件上传的目标路径
          PathStr = Server.MapPath("data/data.mdb")
          PathStr = Left(PathStr, InStrRev(Trim(PathStr), "\") - 1)
          PathStr = Left(PathStr, InStrRev(Trim(PathStr), "\") - 1)
          PathStr = PathStr & "\photo\" & FileUpload1.FileName
          FileUpload1.SaveAs(PathStr)  '执行上传操作
        Catch ex As Exception
          '上传文件失败，提示错误信息，并设置变量 Tag 的值为 1
          ErrStr = "相片上传失败：" & Replace(ex.Message.ToString, "/", "//")
          Label1.Text = Trim(ErrStr)
          Tag = 1
        End Try
        If Tag = 0 Then
          '变量 Tag 等于 0，表示文件上传成功
          StrCnn = System.Configuration.ConfigurationSettings.AppSettings
    ("MM_CONNECTION_STRING_Cnn")  '获取数据库连接字符串
          Cnn = New OleDbConnection(StrCnn)
          Cnn.Open()
          '向数据表 PhotoInfo 中插入所上传的相片信息
          Sql = "insert into photoinfo(picaddr,picname,classid,descr) values('"
    & Trim(FileUpload1.FileName) & "','" & Trim(NameStr) & "'," & Trim(theClass)
    & ",'" & Trim(DescrStr) & "')"
          Cmd = New OleDbCommand(Sql, Cnn)
          Cmd.ExecuteNonQuery()
          '提示成功信息
          ClientScript.RegisterStartupScript(Me.GetType(), "", GetInfo("相片上
    传成功！"))
          Cnn.Close()
          name.Text = ""  '清空相片名称文本框
          Descr.Text = "" '清空相片描述文本框
        End If
      End If
    End Sub
```

在以上代码中，首先调用 FileUpload 控件的 HasFile()方法判断用户是否选择了相片文件。如果没有选择，则不执行任何操作；如果选择了所要上传的相片文件，则先执行文件的上传操作，并通过错误控制来判断文件是否上传成功。如果上传成功，则进一步将相片信息保存至数据库中，并提示相应的成功信息；否则，显示文件上传失败的错误信息，不再执行任何操作。

至此，上传相片页面的功能全部完成，页面预览如图 9.76 所示。

单击【浏览】按钮，可在本地磁盘中选择相片文件。选择文件后，单击【预览】按钮，则可在下方预览用户所选择的相片，如图 9.77 所示。

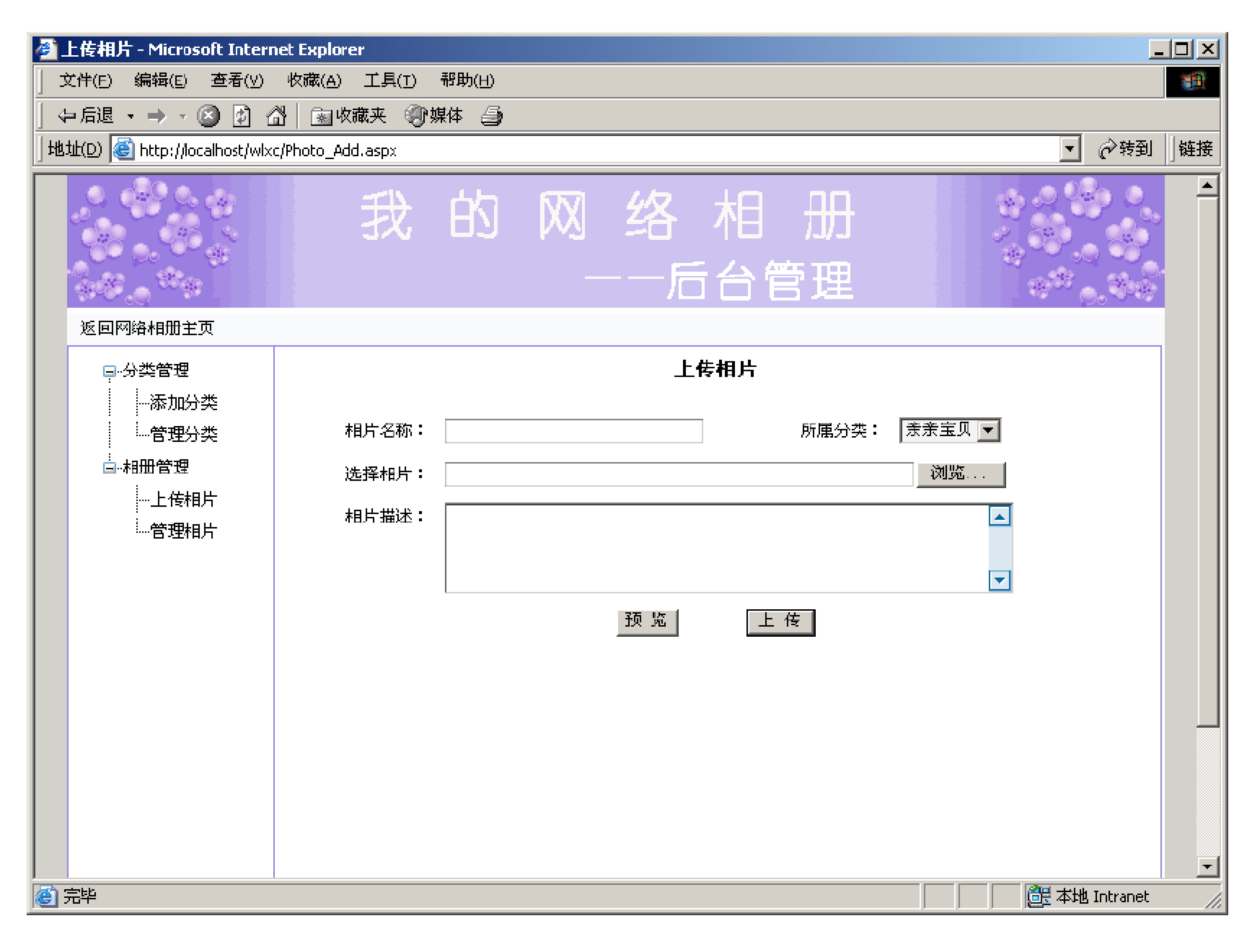


图 9.76　页面预览

图 9.77　相片预览

可以发现，通过 JavaScript 脚本来实现相片的预览，可以提高相片预览的速度。因为它是在客户端执行，而无须经过与服务器的交互。事实上，在使用 ASP.NET 开发 Web 程序时，当与服务器的交互过多而导致程序的响应较慢时，均可考虑采用 JavaScript 脚本来替代 ASP.NET 中的某些服务器行为。

9.3.6　管理相片

管理相片页面主要用于对相片的基本信息进行管理，其功能包括相片信息的查询、预览、编辑和删除等。同时，还应提供对指定相片的留言信息进行管理的相关链接。

(1) 新建一个 ASP.NET VB 类型的页面，将其命名为 Photo_Admin.aspx。选择【修改】|【模板】|【套用模板到页】命令，在弹出的【选择模板】对话框中，选择【后台模板】，将模板应用到本页面。

修改页面标题为“相片管理”，然后在表格 table2 的第 2 个单元格中，添加功能标题“相片管理”。

接下来，将通过数据网格(即 DataGrid 控件)来显示相片的分类信息。在此之前，需要创建一个绑定数据显示的数据集。

(2) 打开【服务器行为】面板，单击+按钮，从弹出的菜单中选择【数据集】命令。在弹出的【数据集】对话框中，单击【高级】按钮，切换至【数据集】对话框的高级模式，如图 9.78 所示。

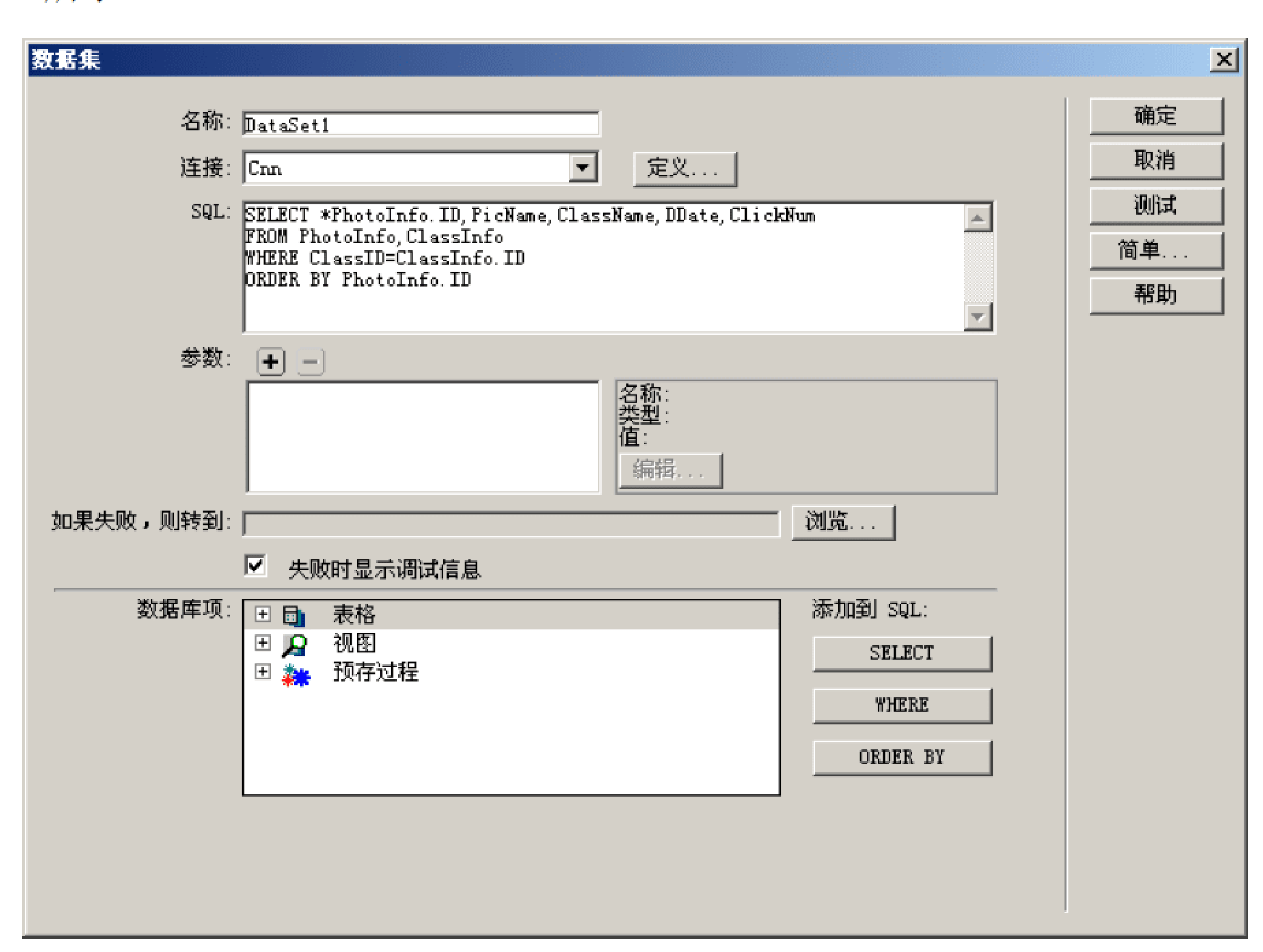


图 9.78　创建数据集 DataSet1

设置数据集名称为 DataSet1，连接选择 Cnn，并在 SQL 文本框中输入如下 SQL 语句：

```
SELECT photoinfo.id,picname,classname,ddate,clicknum FROM
photoinfo,classinfo WHERE classid=classinfo.id ORDER BY photoinfo.id
```

单击【确定】按钮，完成数据集的创建。

(3) 将光标置于功能标题之下，在【服务器行为】面板中，单击+按钮，从弹出的菜单中选择【数据网格】命令，在弹出的【数据网格】对话框中，设置 ID 为 DataGrid1，选择数据集 DataSet1，设置一次显示 10 条记录，从【导航】下拉列表中选择【编号链接到每

一页】，如图 9.79 所示。

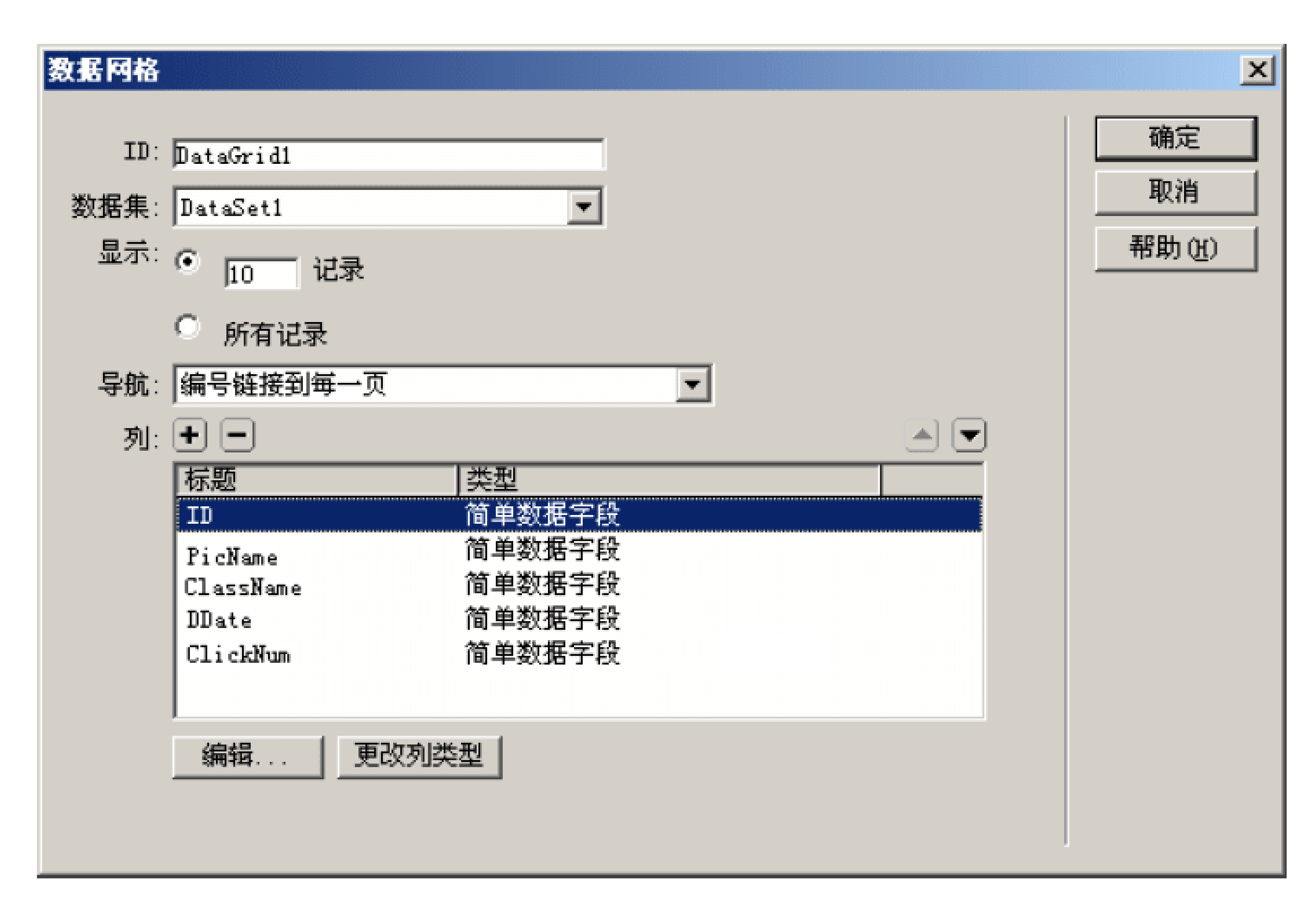


图 9.79 【数据网格】对话框

(4) 单击【添加列】按钮，从弹出的菜单中选择【'编辑'、'更新'、'取消'按钮】命令，在弹出的【编辑、更新、取消按钮列】对话框中，设置【标题】为“编辑”，【按钮类型】为【链接按钮】，【更新表格】为 PhotoInfo，【主键】为 ID，【提交为】为 Integer，如图 9.80 所示。

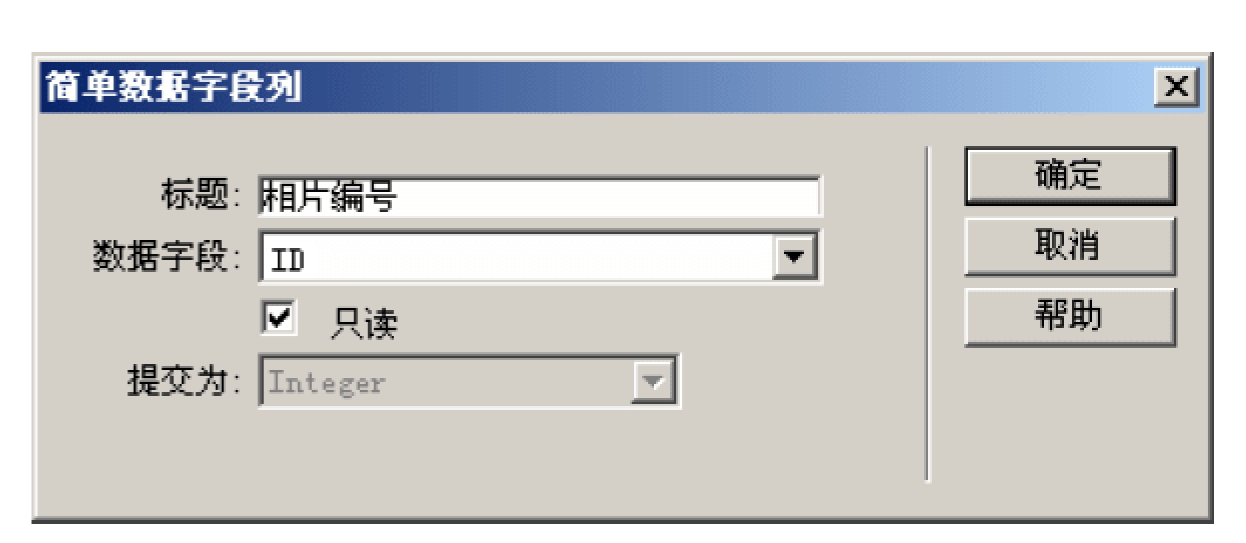


图 9.80 【编辑、更新、取消按钮列】对话框

单击【确定】按钮，完成编辑列的添加。

(5) 再次单击【添加列】按钮，从弹出的菜单中选择【'删除'按钮】，在弹出的【删除按钮列】对话框中设置【标题】为“删除”，【按钮类型】为【链接按钮】，【更新表格】为 PhotoInfo，【主键】为 ID，【提交为】为 Integer，如图 9.81 所示。

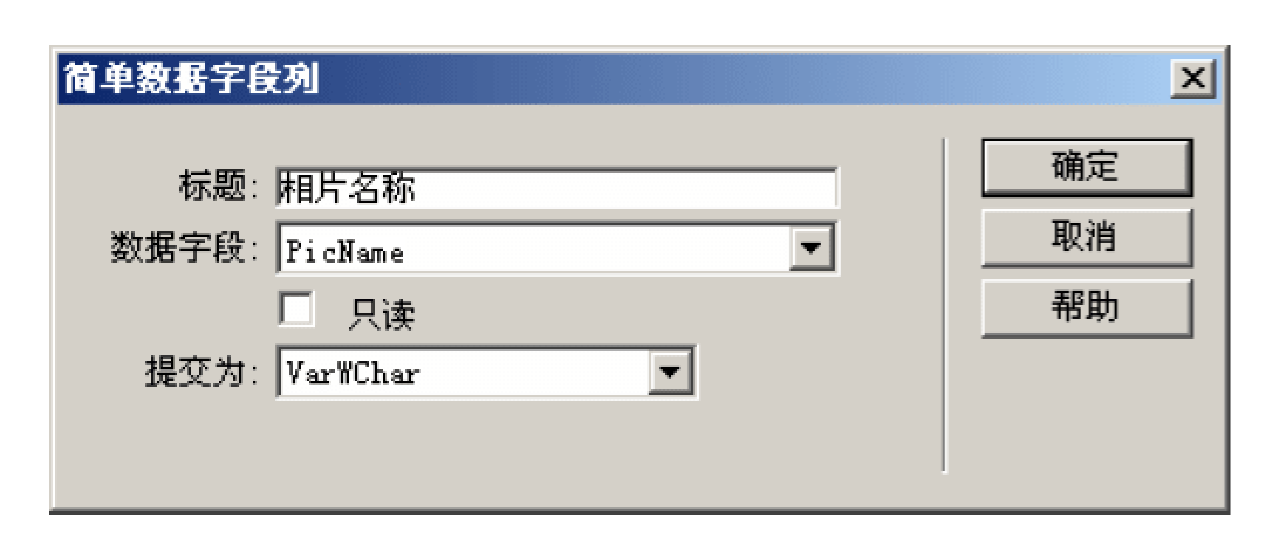


图 9.81 【删除按钮列】对话框

单击【确定】按钮，完成删除列的添加。

(6) 在【列】列表框中，选择 ID 列，单击【编辑】按钮，在弹出的【简单数据字段列】对话框中设置该列的显示标题为“相片编号”，如图 9.82 所示。

图 9.82　设置 ID 列的属性

选择 PicName 列，单击【编辑】按钮，在弹出的【简单数据字段列】对话框中设置该列的显示标题为“相片名称”，同时取消【只读】复选框的选择，如图 9.83 所示。

图 9.83　设置 PicName 列的属性

选择 ClassName 列，单击【编辑】按钮，在弹出的【简单数据字段列】对话框中设置该列的显示标题为“所属分类”，如图 9.84 所示。

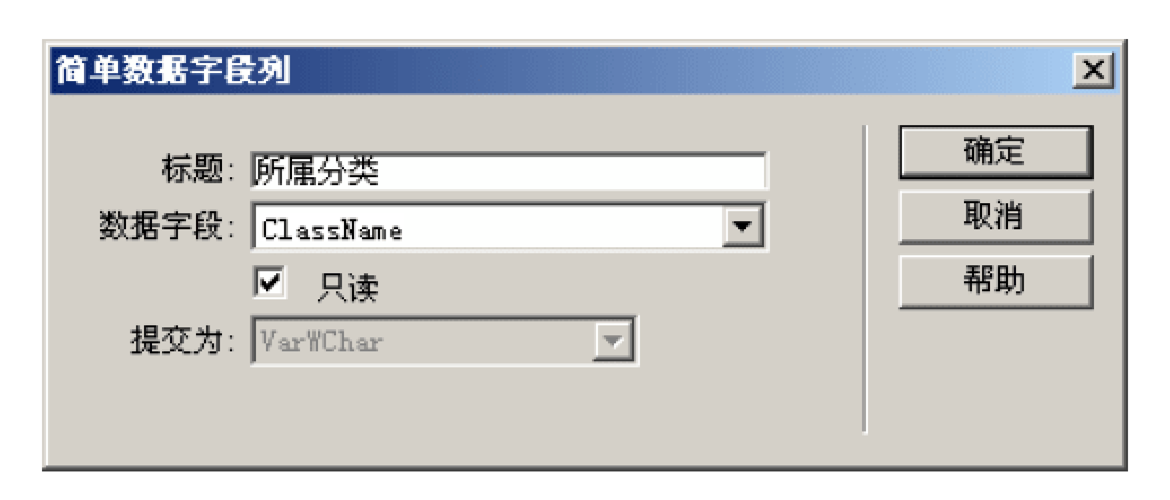


图 9.84　设置 ClassName 列的属性

选择 DDate 列，单击【编辑】按钮，在弹出的【简单数据字段列】对话框中设置该列的显示标题为“上传时间”，如图 9.85 所示。

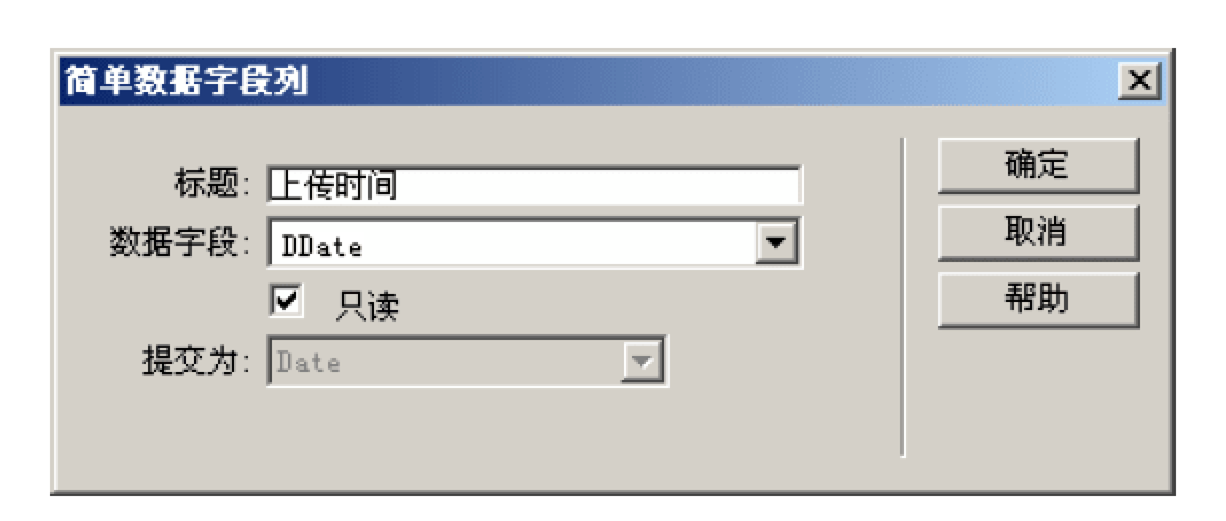


图 9.85　设置 DDate 列的属性

选择 ClickNum 列，单击【编辑】按钮，在弹出的【简单数据字段列】对话框中设置

该列的显示标题为“点击次数”，如图 9.86 所示。

(7) 字段的属性设置完成，接下来添加“管理留言”的链接。单击【添加列】按钮，从弹出的菜单中选择【超级链接】命令，在弹出的【超级链接列】对话框中，设置显示标题为“管理留言”，在【超级链接文本】选项组中设置静态文本为“管理留言”，在【链接页】选项组中设置数据字段为 id，并设置【格式字符串】选项为“photo_ly_admin.aspx?id={0}”，如图 9.87 所示。

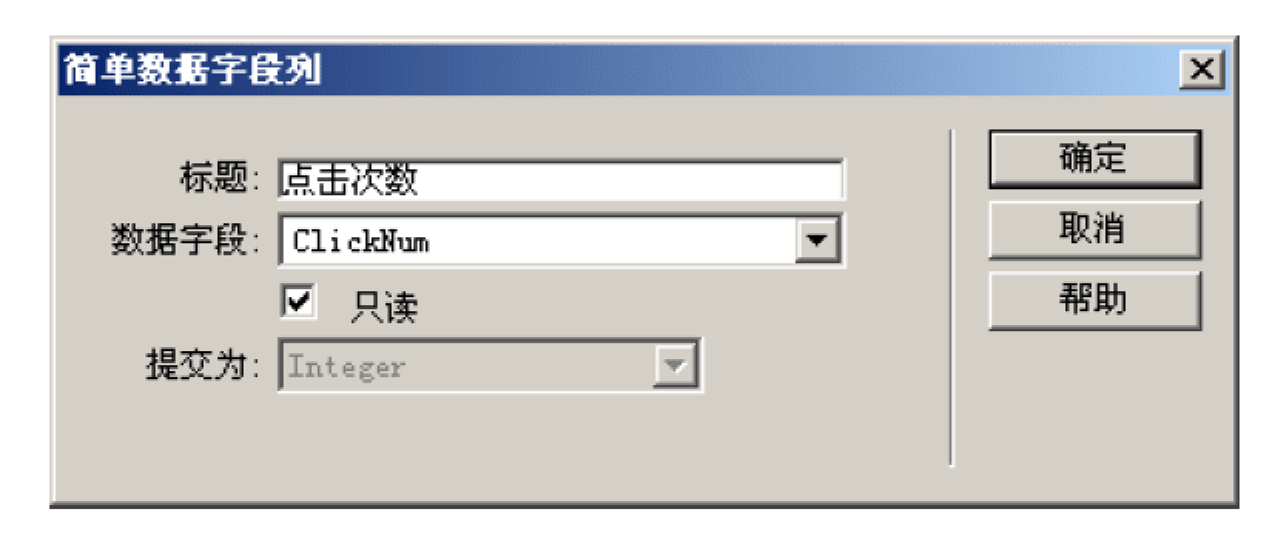


图 9.86　设置 ClickNum 列的属性

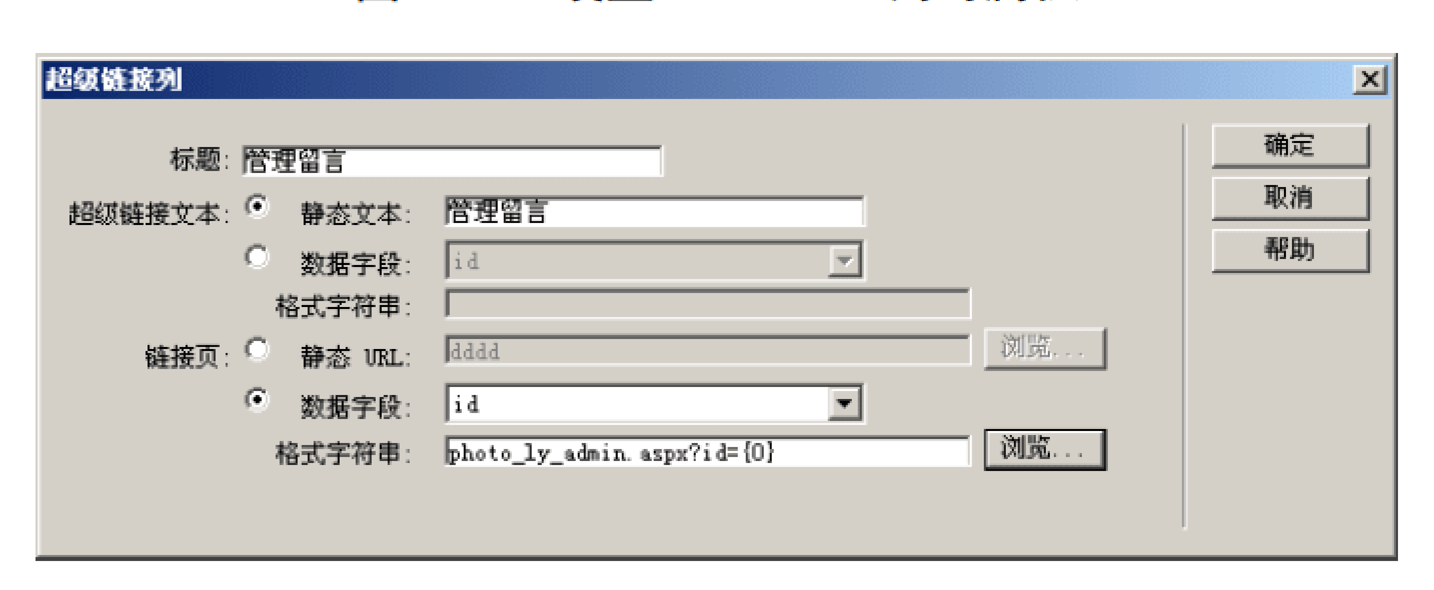


图 9.87　添加超级链接列

(8) 单击【确定】按钮，完成超级链接列的添加。

此时，在数据网格中所需显示的各项列均设置完成，如图 9.88 所示。

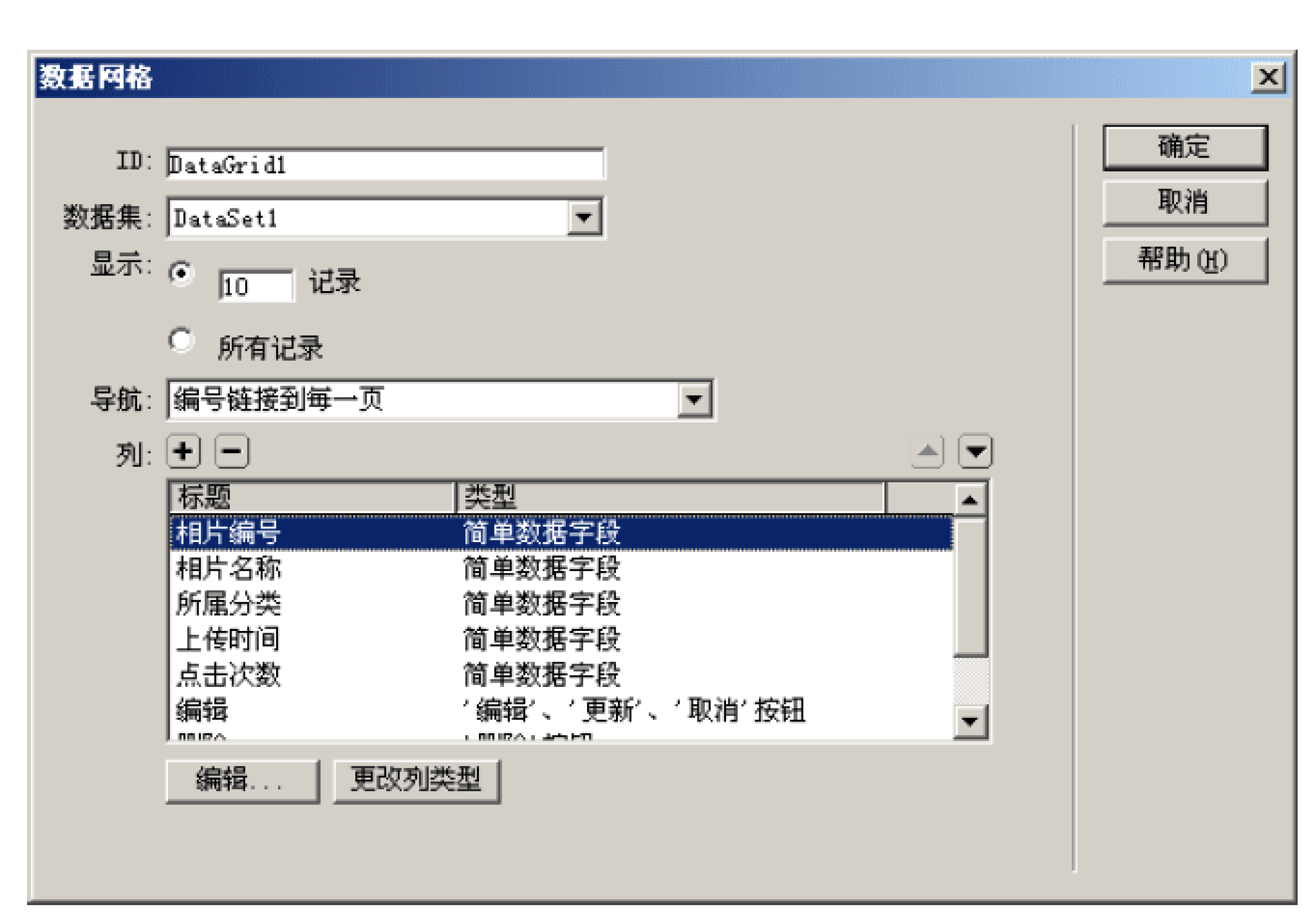


图 9.88　数据网格中列的设置

单击【确定】按钮，完成数据网格的初步创建。

事实上，在数据网格中还需添加一个按钮列，即【预览】列。为什么没有在以上的可

视化操作中进行添加呢？这是由于在 Dreamweaver 8 的数据网络中，不支持可视化地添加按钮列(在添加列的选项中，可以看见并无“按钮列”一项)。因此，只能在代码中进行手动添加。

(9) 切换至【代码】视图，定位到 DataGrid 控件的定义代码中，在【删除】列和【管理留言】列中添加【预览】列的代码，如图 9.89 所示。

```
<asp:ButtonColumn
        ButtonType="LinkButton"
        CommandName="Delete"
        HeaderText="删除"
        Text="删除"
        Visible="True"/>
<asp:ButtonColumn
        ButtonType="LinkButton"
        CommandName="ViewPhoto"
        HeaderText="预览"                 插入“预览”列
        Text="预览"
        Visible="True"/>
<asp:HyperLinkColumn DataNavigateUrlField="id"
DataNavigateUrlFormatString="photo_ly_admin.aspx?id={0}"
        Visible="True"
        HeaderText="管理留言"
        Text="管理留言"/>
</Columns>
```

图 9.89　添加【预览】列的代码

(10) 对数据网格的显示作进一步调整。在【设计】视图中，选择所创建的数据网格并右击，从弹出的快捷菜单中选择【编辑标签】命令，在弹出的【标签编辑器】对话框中选择【布局】分类，设置其宽度为 620 像素，单元格边距为 3，单元格间距为 0，边框宽度为 1。

(11) 在数据网格的下方插入一个 ID 为 Image1 的 Image 服务器控件，用于预览相片。选择该控件并右击，从弹出的快捷菜单中选择【编辑标签】命令，在弹出的【标签编辑器】对话框中设置【布局】项下的宽度值为 200 像素，同时取消【样式信息】项下的【可见】复选框。即在用户单击【预览】按钮之前，该 Image 控件为隐藏状态；当用户单击【预览】按钮时，再将其显示。

(12) 来看一下数据网格中各种功能按钮的事件。对于编辑和管理留言的操作，均可采用其默认的行为而无须更改；对于删除操作，如果采用数据网格所提供的默认行为，则仅是删除相片信息。但是，为了保证数据的完整性，在删除相片信息的同时还应删除该相片的所有留言信息。因此，需要手动设置删除操作的事件代码。而对于预览操作，其按钮列为手动添加，Dreamweaver 8 的数据网格设置中并无其相应的默认行为，需要手动添加其相关的事件。

在 Dreamweaver 8 所提供的数据网格的标签编辑器中，提供了数据网格的各种事件。其中，【删除】按钮列的操作所对应的默认事件为 OnItemCommand，而【预览】按钮列的执行操作必须设置在 OnPreRender 事件中。事实上，包括【删除】列在内的所有按钮列的操作均会触发 OnPreRender 事件，且其顺序在 OnItemCommand 事件之前。因此，完全可以删除在【删除】按钮列所对应的默认事件 OnItemCommand 中定义的操作，而在 OnPreRender 事件中添加自定义的事件代码，同时处理【删除】列和【预览】列的操作。

提示： 在标签编辑器中所列的事件与实际在代码中所生成的 DataGrid 事件并不一致，例如标签编辑器中的 OnItemCommand 事件在代码中变为 OnDeleteCommand 事件，而编

辑器中的 OnPreRender 事件在代码中又变为 OnItemCommand 事件，这是 Dreamweaver 8 在处理数据网络时的一个 Bug。最终，应以代码中的事件为准。

(13) 将视图切换至【设计】视图，选择所创建的数据网格并右击，从弹出的快捷菜单中选择【编辑标签】命令，在弹出的【标签编辑器】对话框中，选择【事件】|OnItemCommand，将其默认的事件名 DataSet1. OnDataGridDelete 删除；选择【事件】|OnPreRender，在其中添加自定义的事件 DataGrid1_Mod，如图 9.90 所示。

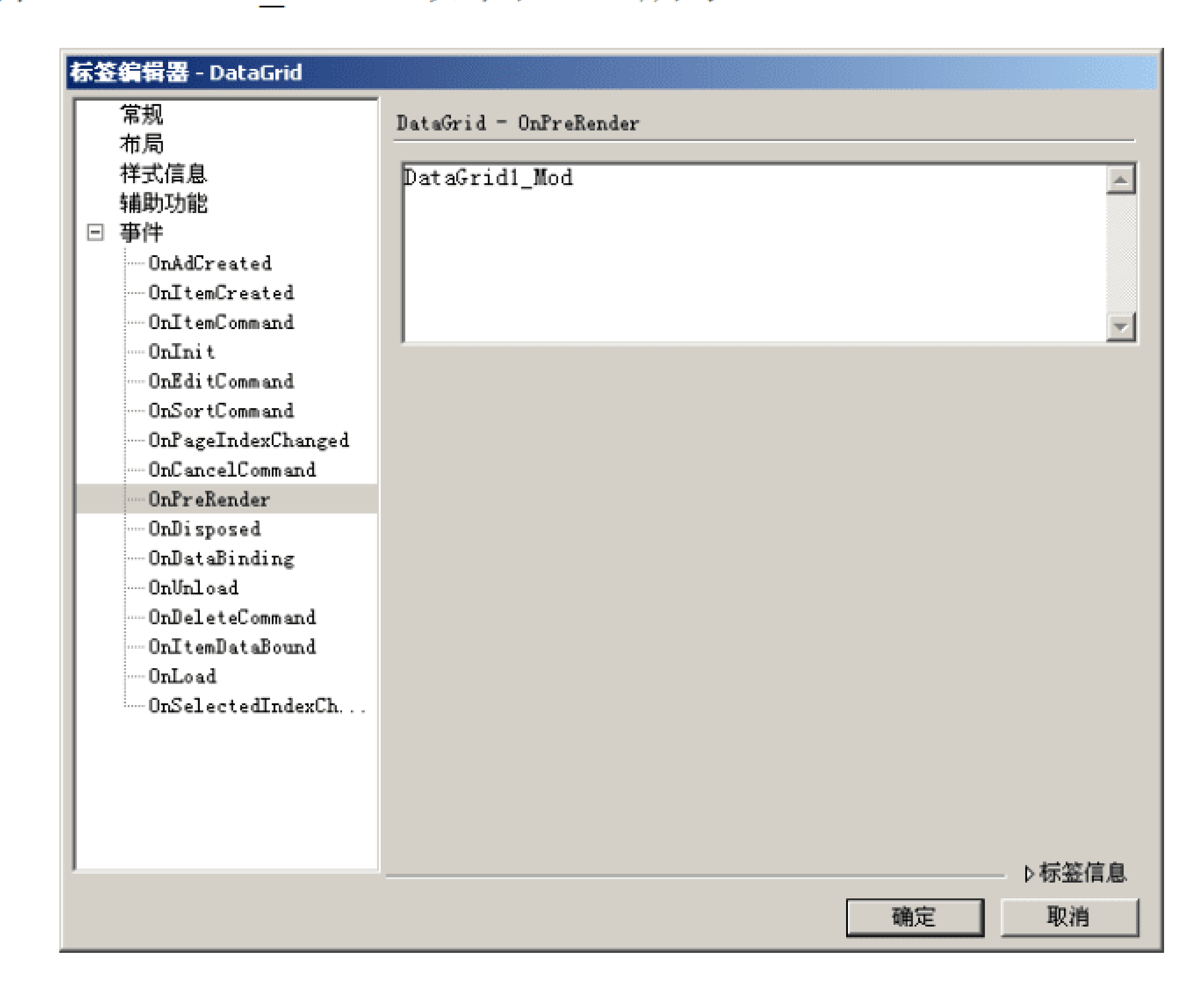


图 9.90　添加 OnPreRender 事件

(14) 单击【确定】按钮，完成数据网格的事件设置。切换至【代码】视图，添加 DataGrid1_Mod 事件代码，如下所示：

【示例代码】

```
Sub DataGrid1_Mod(ByVal Sender As Object, ByVal E As
DataGridCommandEventArgs)
  '获取当前所要操作的相片 ID
  Dim codestr As String = E.Item.Cells(0).Text
  Dim DataR As OleDbDataReader
  Dim Cnn As OleDbConnection
  Dim Cmd As OleDbCommand
  Dim StrCnn As String
  Dim Sql As String
  '获取数据库连接字符串
  StrCnn =
System.Configuration.ConfigurationSettings.AppSettings("MM_CONNECTION_ST
RING_Cnn")
  '根据触发当前事件的按钮列的 CommandName 属性，判断应执行何种操作
  If CType(e.CommandSource, LinkButton).CommandName = "Delete" Then
    'CommandName 属性为“Delete”，执行删除操作
    Cnn = New OleDbConnection(StrCnn)
    Cnn.Open()
    '删除指定相片的留言信息
```

```
        Sql = "delete from photoly where photoid=" & trim(codestr)
        Cmd = New OleDbCommand(Sql, Cnn)
        Cmd.ExecuteNonQuery()
        '删除指定的相片信息
        Sql="delete from photoinfo where id=" & trim(codestr)
        Cmd=New OleDbCommand(Sql,Cnn)
        Cmd.ExecuteNonQuery()
        Cnn.Close()  '关闭数据库连接
        '页面跳转至本页面，刷新数据的显示
        Response.Redirect("Photo_Admin.aspx")
      Else If CType(e.CommandSource, LinkButton).CommandName = "ViewPhoto" Then
        'CommandName 属性为“ViewPhoto”，执行预览操作
        '查询所要预览的相片的图像路径和相片名称
        Sql="Select picname,('~/photo/'+picaddr) as pic_addr from photoinfo
    where id=" & Trim(codestr)
        Cnn = New OleDbConnection(StrCnn)
        Cnn.Open()  '打开数据库连接
        Cmd=New OleDbCommand(Sql,Cnn)
        DataR=Cmd.ExecuteReader()
        If DataR.Read() Then
          '设置 Image 控件的 ImageUrl 属性
          Image1.ImageUrl = DataR("pic_addr")
          '设置 Image 控件的 AlternateText 属性
          Image1.AlternateText = DataR("picname")
          '将 Image 控件置为可见
          Image1.Visible = True
        End if
        Cnn.Close()  '关闭数据库连接
      End If
    End Sub
```

在 DataGrid1_Mod 事件中，首先依据 CType(e.CommandSource, LinkButton)获取触发事件的源，并将其转化为 LinkButton 类型；然后获取其 CommandName 属性，并依据该属性值来判断所要执行的操作。当 CommandName 属性值为 Delete 时，表示触发事件的按钮为【删除】按钮，应执行删除相片信息操作；当 CommandName 属性值为 ViewPhoto 时，表示触发事件的按钮为【预览】按钮，应执行预览相片操作。

至此，管理相片的页面功能全部完成，页面预览如图 9.91 所示。

单击【编辑】按钮，可对指定相片的相片名称信息进行编辑，此时【相片名称】栏将显示为文本框，而【编辑】按钮也将改变为【更新】和【取消】按钮；单击【删除】按钮，则可删除指定的相片信息；单击【预览】按钮，可在数据网格的下方显示该相片的图像，如图 9.92 所示。

对于“管理留言”功能，这里仅是创建了一个目标页面为 Photo_LY_Admin.aspx 的链接，该页面的功能实现将在下一节进行介绍。

提示： 在此页面中，必须导入命名空间 System.Data 和 System.Data.OleDb。

图 9.91　页面预览

图 9.92　相片预览

9.4　相册留言管理

本节讲解相片留言信息的管理功能的实现方法，在该管理功能中，首先提供对指定相片的留言信息的浏览，然后在此基础上提供对留言的删除功能。

新建一个 ASP.NET VB 类型的页面，将其命名为 Photo_LY_Admin.aspx。选择【修改】|【模板】|【套用模板到页】命令，在弹出的【选择模板】对话框中，选择【后台模板】，将模板应用到本页面。同时，将页面标题设置为“留言管理”。

在页面结构的设计上，本页面与查看相片功能的页面结构较为相似，两者均包括相片信息显示和留言信息浏览两部分。不同的是，本页面在显示相片信息的表格中将添加【删除所有留言】按钮，而在留言信息浏览中将针对每条留言添加【删除】按钮。

(1)　首先，来看相片信息的显示。在表格 table2 的第 2 个单元格中，插入一个 6 行 3 列的表格，合并第 1 列的 6 个单元格，并在合并的单元格中插入一个 Image 服务器控件，设置其宽度为 250，同时在其单元格中输入相应的文本，并在第 5 行的第 2 列中插入按钮控件，如图 9.93 所示。

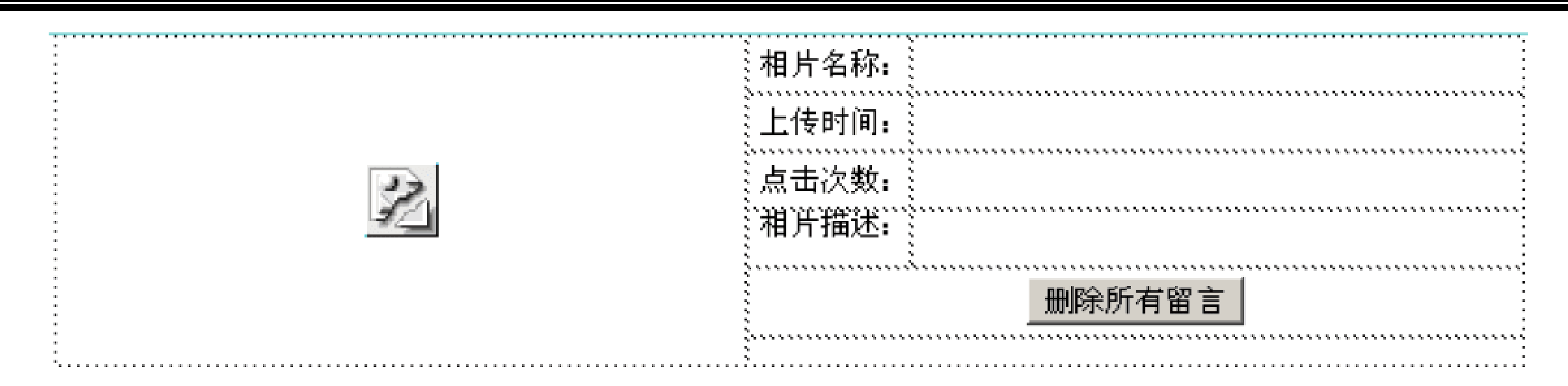


图 9.93　插入的嵌套表格

其中，相片名称、上传时间、点击次数、相片描述以及 Image 控件的 ImageUrl 属性均需绑定动态文本。在此之前，需要创建一个数据集。

(2) 在【应用程序】面板组中，切换至【服务器行为】面板，单击按钮，在弹出的菜单中选择【数据集】命令。在弹出的【数据集】对话框中，单击【高级】按钮，切换至数据集的高级模式。在高级模式下，单击【添加参数】按钮，在弹出的【编辑参数】对话框中，设置参数名称为“@ID”，类型为 Integer，如图 9.94 所示。

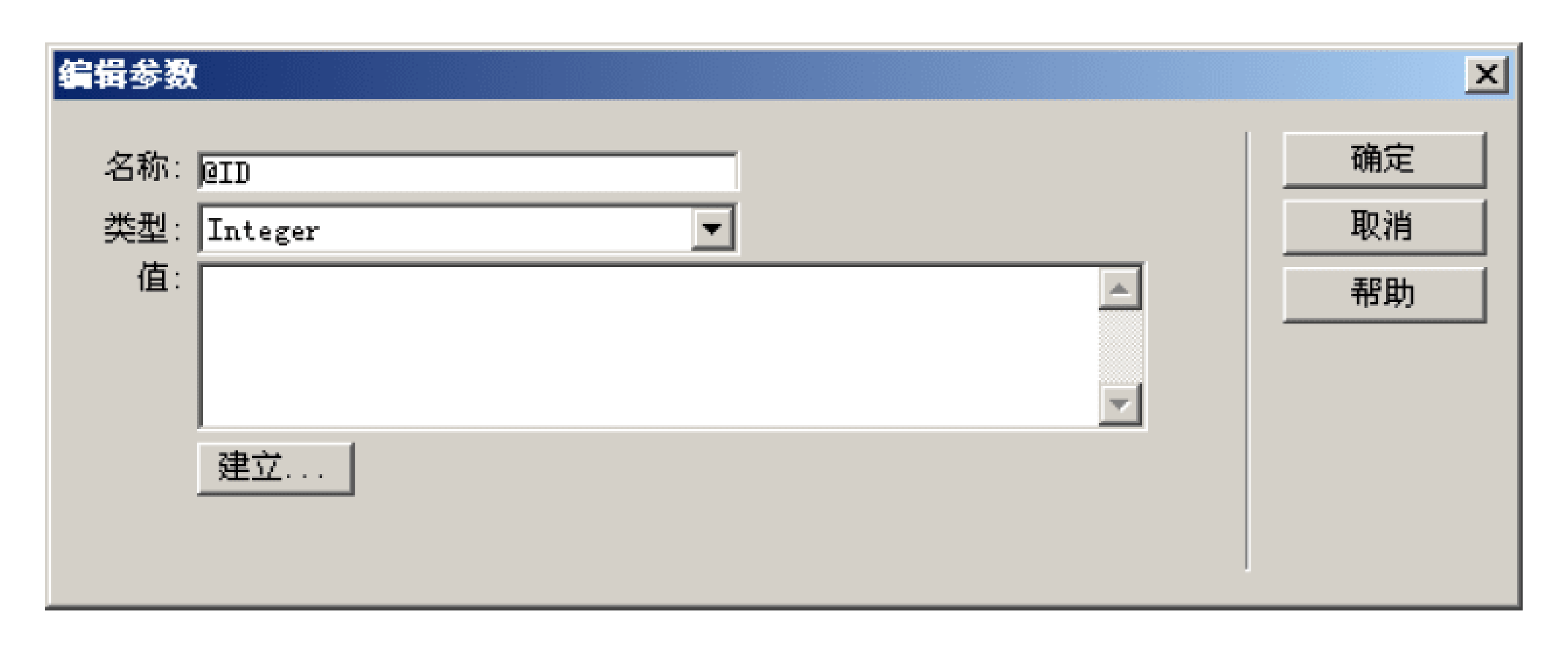


图 9.94　【编辑参数】对话框

单击【建立】按钮，在弹出的【生成值】对话框中，设置参数的值为 URL 参数 ID，如图 9.95 所示。

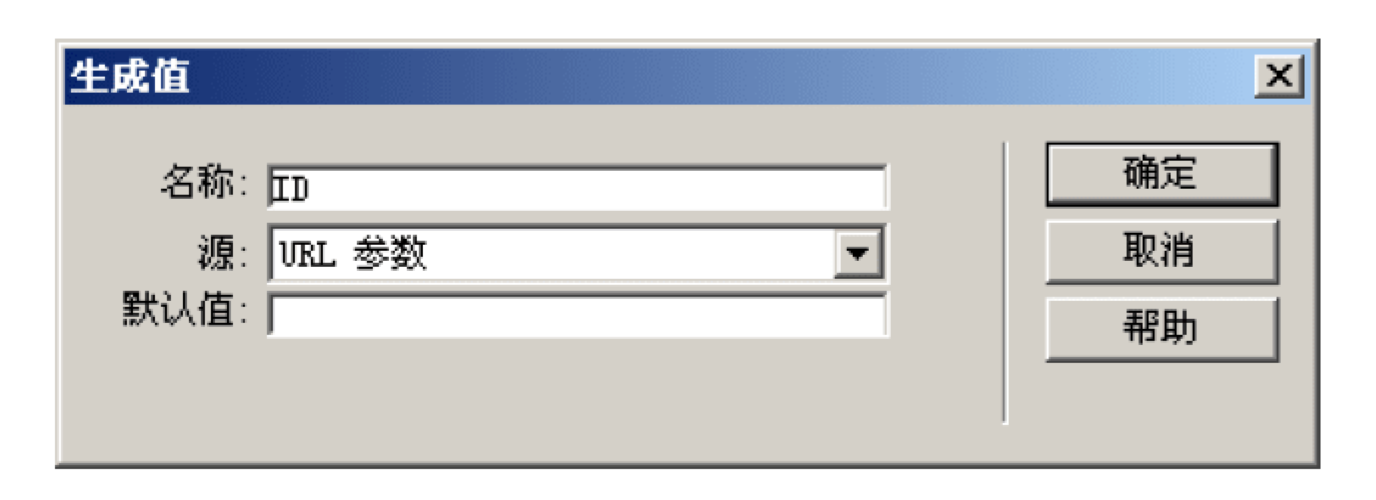


图 9.95　【生成值】对话框

(3) 单击【确定】按钮，返回【数据集】对话框。设置数据集名称为 DataSet1，连接为 Cnn，同时在 SQL 文本框中输入以下 SQL 语句：SELECT *,('~/photo/'+picaddr) as pic_addr from photoinfo where id=?，如图 9.96 所示。

单击【确定】按钮，完成数据集的创建。此数据集将根据页面参数 ID 的值，返回对应的相片信息。其中，字段 Pic_Addr 表示相片所对应的图像路径。

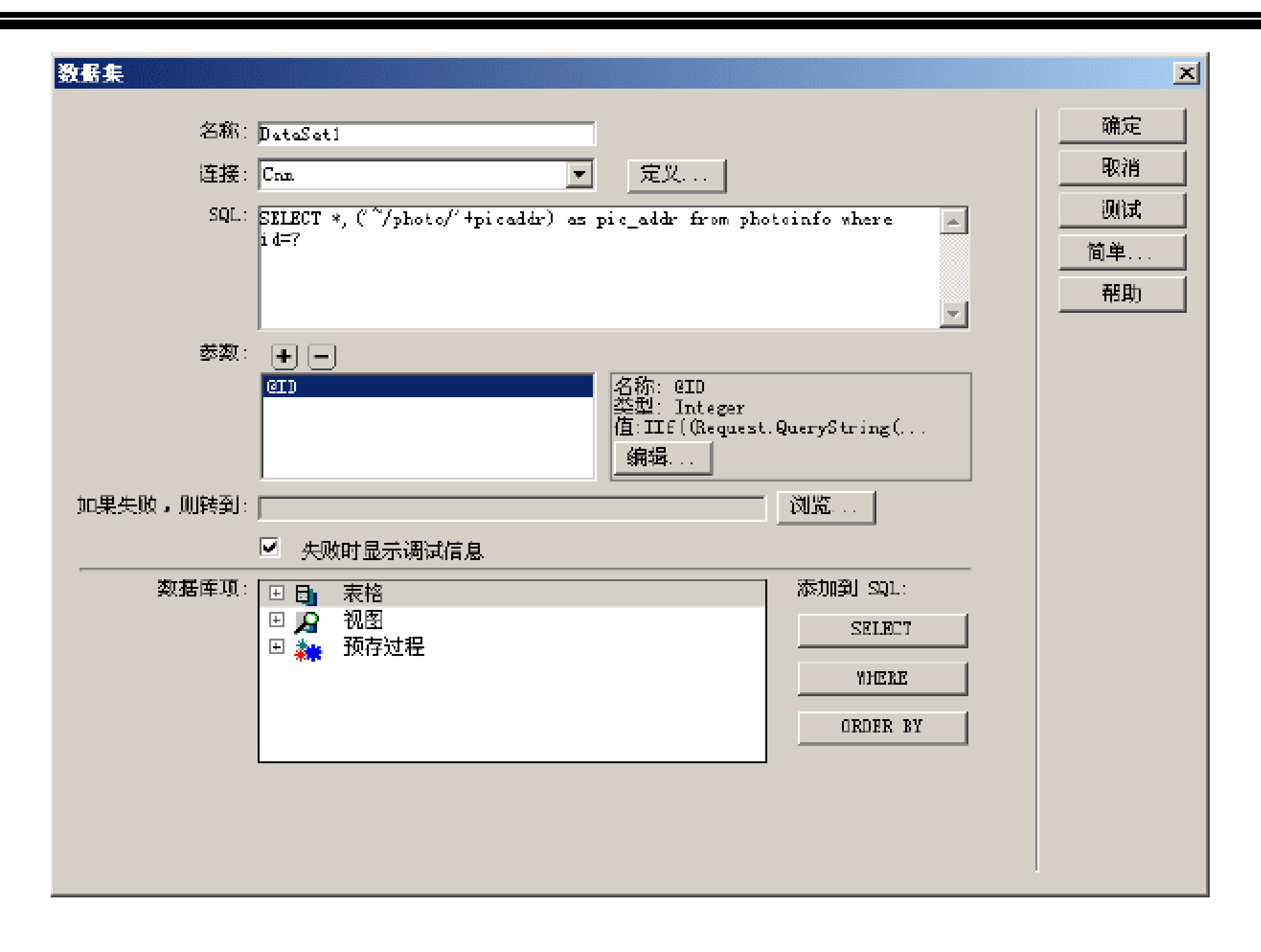


图 9.96　创建数据集 DataSet1

(4)　打开【绑定】面板，选择数据集 DataSet1，并将其展开，如图 9.97 所示。

拖动字段 PicName 至文本“相片名称”后的单元格中，并在【属性】工具栏中设置该动态文本的字体颜色值为“#3399CC”，加粗显示。与此类似，依次拖动字段 DDate、ClickNum、Descr 至文本“上传时间”、“点击次数”、“相片描述”后的单元格中，并设置其相应的字体属性。同时，设置 Image 控件的 ImageUrl 属性的值为<%# DataSet1.FieldValue ("pic_addr", Container) %>，从而完成指定相片信息的数据绑定和显示。

图 9.97　【绑定】面板

(5)　选择【删除所有留言】按钮并右击，从弹出的快捷菜单中选择【编辑标签】命令，在弹出的【标签编辑器】对话框中，选择【事件】|OnClick 事件，在右侧的文本框中输入 DelAll_Click，如图 9.98 所示。

此操作是设置当单击【删除所有留言】按钮时所执行的事件，其事件代码将在后面讲述。

接下来，实现留言信息的【浏览】按钮及【删除】按钮的添加。

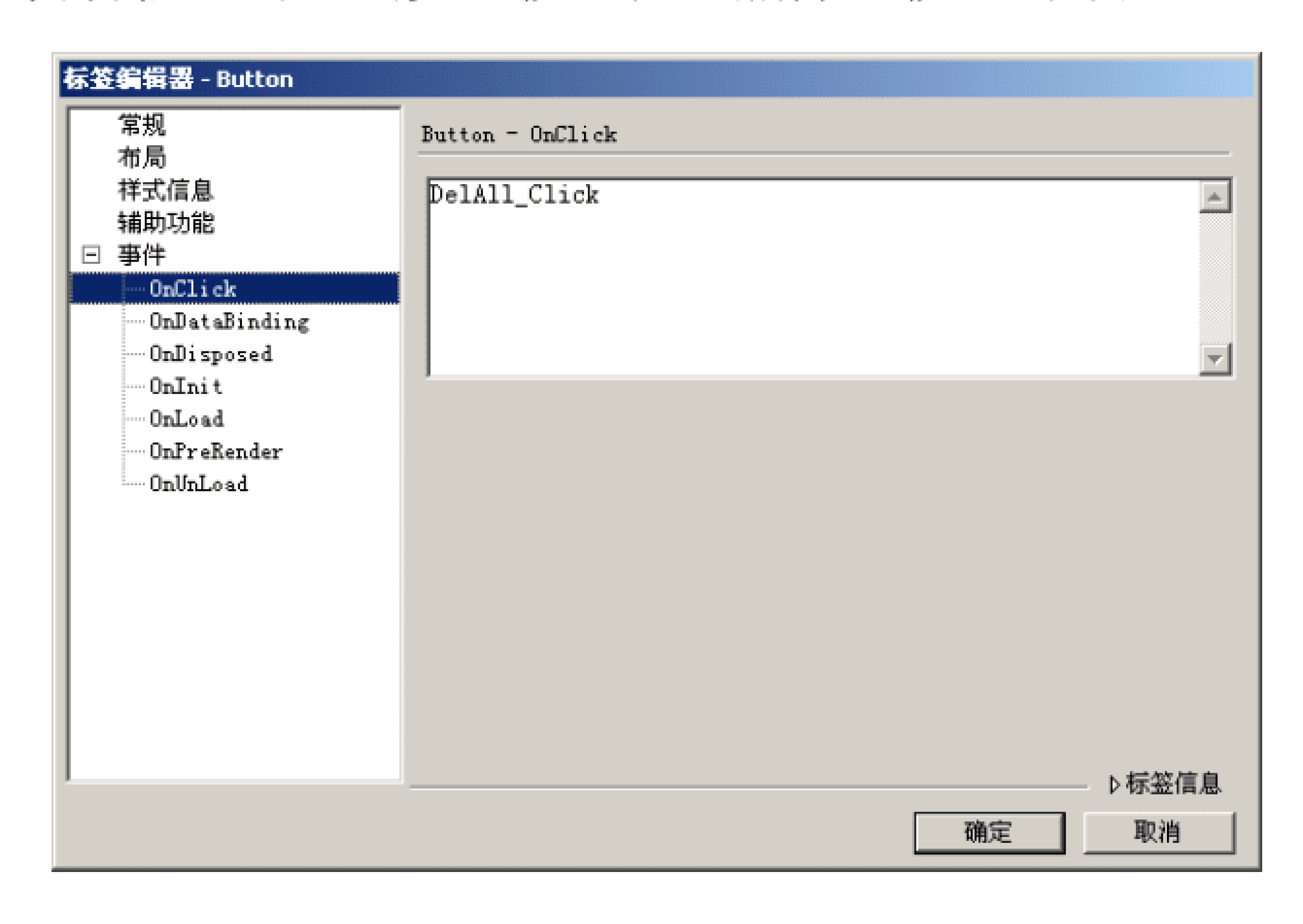


图 9.98　设置按钮的 OnClick 事件

(6) 创建绑定显示留言信息的数据集。在【应用程序】面板组中，切换至【服务器行为】面板，单击按钮，从弹出的菜单中选择【数据集】命令。在弹出的【数据集】对话框中，设置数据集名称为 DataSet2，连接选择 Cnn，表格选择 PhotoLY，筛选条件设置为字段 PhotoID 等于 URL 参数 ID，如图 9.99 所示。

图 9.99　创建数据集 DataSet2

(7) 单击【确定】按钮，完成数据集的创建。这里创建的数据集所返回的记录为指定相片的所有留言信息。

下面创建单条留言信息的数据绑定显示。

(8) 在相片信息表格的下方，再次插入一个 2 行 3 列的表格，将第 1 行的背景色值设置为#0000FF，字段颜色值设置为#FFFFFF，然后在第 1 行的第 1 个单元格中输入“留言人：”，第 2 个单元格中输入“留言时间：”，第 3 个单元格中插入【删除】按钮控件，最后再合并第 2 行的 3 个单元格，如图 9.100 所示。

图 9.100　插入表格

其中，在插入【删除】按钮时，设置其 ID 为 Button1，文本为“删除”，命令名称为 DelLy，命令参数为<%# DataSet2.FieldValue("ID", Container) %>，如图 9.101 所示。

(9) 在【代码】视图中，添加 Button 控件的 OnCommand 属性为 Del_Click，即单击按钮的执行事件。

(10) 打开【绑定】面板，选择数据集 DataSet2，并将其展开，如图 9.102 所示。

(11) 选择字段 LYName 并将其拖至文本“留言人：”之后，选择并拖动字段 LYTime 至文本“留言时间：”之后，选择并拖动字段 LYContent 至表格第 2 行的单元格中。同时在【服务器行为】面板中选择并双击其中的“动态文本(DataSet2.LYContent)”，在弹出的动态文本】对话框中，设置其格式为【编码-HTML 编码格式】，如图 9.103 所示。

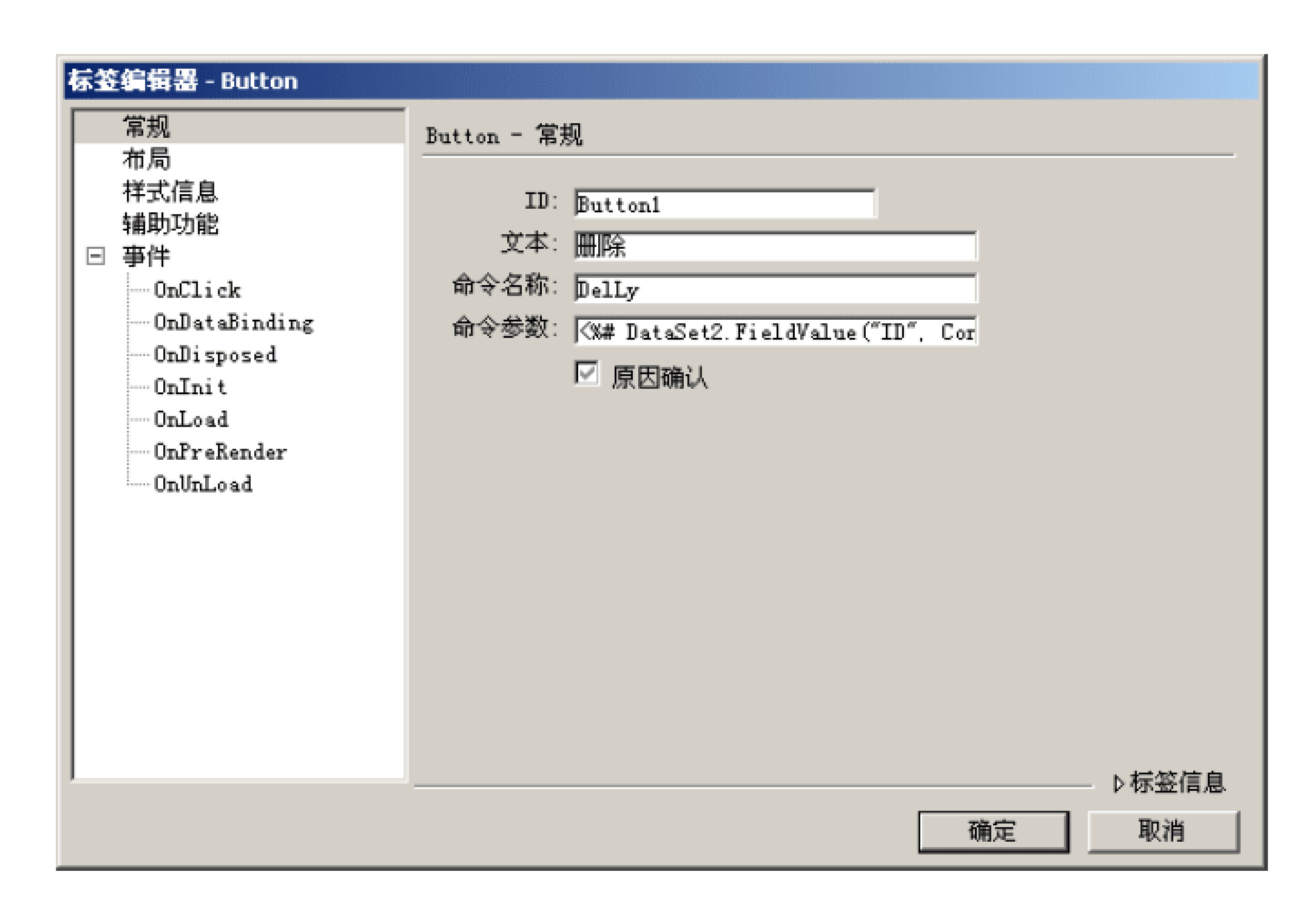


图 9.101　插入【删除】按钮

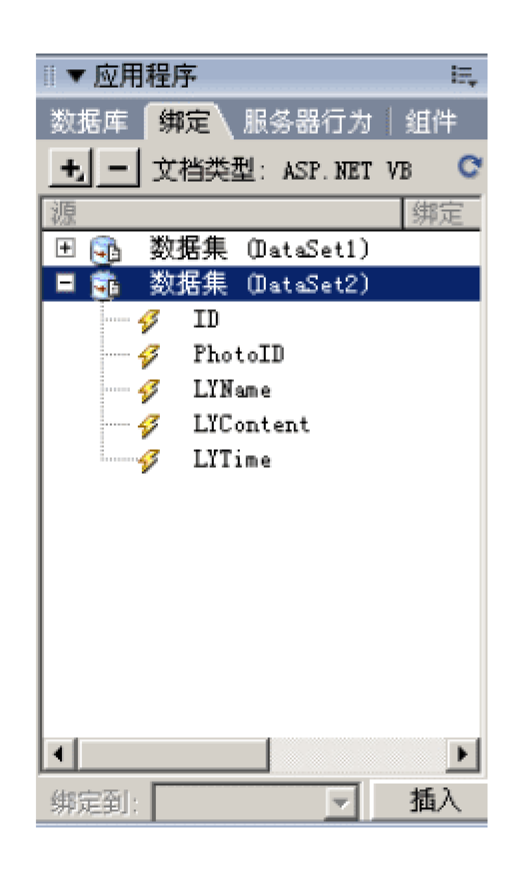


图 9.102　【绑定】面板　　图 9.103　设置动态文本格式

单击【确定】按钮，完成单条留言数据的绑定显示。

(12) 选定显示留言信息的整个表格，在【服务器行为】面板中单击＋按钮，在弹出的快捷菜单中选择【重复区域】命令，在弹出的【重复区域】对话框中选择 DataSet2 作为数据集，设置显示所有记录，如图 9.104 所示。

图 9.104　设置重复区域

单击【确定】按钮，完成重复区域的设置。

至此，留言信息的浏览数据绑定完成了，剩下的工作就是实现【删除】按钮的单击操作。

(13) 首先，来看一下【删除所有留言】按钮的单击事件，其代码如下：

【示例代码】

```
Sub DelAll_Click(ByVal Sender As Object, ByVal E As EventArgs)
  Dim Cnn As OleDbConnection
  Dim Cmd As OleDbCommand
  Dim StrCnn As String
  Dim Sql As String
  '获取当前显示留言的相片 ID
  Dim PhotoId as string=request.QueryString("id")
  '获取数据库连接字符串
  StrCnn =
System.Configuration.ConfigurationSettings.AppSettings("MM_CONNECTION_ST
RING_Cnn")
  Cnn = New OleDbConnection(StrCnn)
  Cnn.Open()
  '删除指定相片下的所有留言信息
  Sql="Delete from photoly where photoid=" & trim(photoid)
  cmd=new OleDbCommand(sql,cnn)
  Cmd.ExecuteNonquery()
  Cnn.close()  '关闭数据库连接
  '重定向本页面
  Response.Redirect("photo_ly_admin.aspx?id=" & request.QueryString("id"))
    End Sub
```

当单击【删除所有留言】按钮时，将删除当前相片下的所有留言信息，并将页面重定向本页面，刷新数据的显示。

再来看一下【删除】按钮的单击事件，其代码如下：

【示例代码】

```
Sub Del_Click(Byval Sender As Object ,Byval E As CommandEventArgs)
  Dim Cnn As OleDbConnection
  Dim Cmd As OleDbCommand
  Dim StrCnn As String
  Dim Sql As String
  '获取所要删除的留言 ID
  Dim theID as String=E.CommandArgument
  '获取连接数据库字符串
  StrCnn =
System.Configuration.ConfigurationSettings.AppSettings("MM_CONNECTION_ST
RING_Cnn")
  Cnn = New OleDbConnection(StrCnn)
  Cnn.Open()
  '删除指定的留言信息
  Sql="Delete from photoly where id=" & trim(theid)
  cmd=new OleDbCommand(sql,cnn)
  Cmd.ExecuteNonquery()
  Cnn.close()
  '重定向本页面
  Response.Redirect("photo_ly_admin.aspx?id=" &
request.QueryString("id"))
End sub
```

在定义【删除】按钮时，设置了按钮的 CommandArgument 属性为绑定数据集 DataSet2 中的字段 ID 的值，即留言信息所对应的 ID 值。因此，这里可通过 E.CommandArgument 来获取所要删除的留言信息 ID，并执行删除操作。

需要注意的是，由于【删除】按钮处于 Repeater 数据控件之中，因此这里必须取消页面的事件验证，否则在 ASP.NET 2.0 下运行此页面将会导致页面异常，如图 9.105 所示。

```
<%@ Page Language="VB" ContentType="text/html" ResponseEncoding="gb2312"
EnableEventValidation="false"%>
<%@ Register TagPrefix="MM" Namespace="DreamweaverCtrls" Assembly=
"DreamweaverCtrls,version=1.0.0.0,publicKeyToken=836f606ede05d46a,culture=neutral
" %>
```

图 9.105　取消页面的事件验证

至此，留言管理页面的功能全部完成，页面预览效果如图 9.106 所示。

图 9.106　页面预览效果

9.5　习　　题

(1) 在查看相片功能中，进一步提供对相片的放大功能，即单击相片，打开一个单独的页面，以原尺寸显示该图像。

(2) 在对相片留言的管理功能中，提供对相片留言的屏蔽功能和回复功能。前者，不删除留言，仅屏蔽不让其显示；后者，允许管理员对他人的留言进行回复。

(3) 了解在 ASP.NET 中对文件的相关操作，在删除相片的功能中同时删除相片所对应的文件。

第 10 章　同　学　录

随着互联网的发展，同学录也逐渐成为很多用户上网必去的一个居所。同学录是一个建立在同学用户基础上的虚拟社区，它以班级或学校为单位，将众多的人聚集在一起。它针对曾经是学生或目前仍是学生的人，提供了一个与同窗好友交流的平台。在同学录中，能够查询老同学目前的情况，也能够与同学进行相互交流、沟通，重忆过去的校园生涯。

本章将介绍如何通过 ASP.NET 来构建一个属于自己的同学录。

10.1　系 统 分 析

10.1.1　系统功能

1. 用户管理

在同学录中，用户是以同学的身份存在，他又属于自己的学校和班级。因此，用户只有注册并登录后，才能正常访问同学录系统。在用户管理中，包括用户注册、用户登录、找回密码以及修改个人资料等功能，其实现原理与第 8 章中介绍的会员管理中的相应功能基本相同。

2. 班级留言

在用户所属的班级中，允许浏览班级中其他同学的所有留言信息并发表自己的留言。

3. 班级相册

在用户所属的班级中，可浏览班级中其他同学上传的相片。针对具体的相片，可查看并发表相应的评论。同时，用户也可上传自己的相片。

4. 班级通讯录

在用户所属的班级中，可浏览班级其他同学的通讯录，其数据来源取自各位同学的注册信息。

5. 班级管理

对于每一个班级，允许存在一个班级管理员。班级管理员可对班级的所有成员进行管理，包括通过成员的加入，或将其踢出班级。如果某班级尚无管理员，其任何一个成员均可申请担任班级的管理员。在申请管理员之后，其状态处于待定。一旦班级其他成员有一半以上的人同意该申请，则申请者正式成为该班级的管理员，可行使其相应的权力。而一旦班级其他成员有一半以上的人均不同意该申请，则此次申请失败，其他成员可继续申请担任该班级的管理员。

6. 搜索学校/班级

可通过学校名称、学校类型、所属省份等条件来搜索相应的学校。在搜索结果中，单击相应的学校，可查看该校下面所有注册的班级信息。同时，用户可申请加入查找到的指定班级。如果未能搜索到相应的学校，用户可注册新的学校。如果在指定的学校下，没有自己所在的班级，用户可注册新的班级。

7. 搜索同学

通过姓名可在整个同学录内搜索同学信息。

10.1.2　数据库的建立

本系统使用 Microsoft Access 2000 类型的数据库，其数据库的文件名为 Data.Mdb，在该数据库中添加了 ClassInfo(班级基本信息表)、LYInfo(班级留言信息表)、PhotoInfo(班级相片信息表)、Photo_PL(相片评论信息表)、SchoolInfo(学校基本信息表)、Sys_Province(省份参数信息表)、UserClass(用户班级关联表)和 UserSheet(用户信息表)8 个数据表。

1. 班级基本信息表(ClassInfo)

ClassInfo 数据表主要用于存储班级的基本信息，其表结构见表 10.1。

表 10.1　ClassInfo 数据表

字 段 名	数据类型	字段描述	字段大小	默 认 值	主　键
ID	自动编号	班级编号	长整型		P
ClassName	文本	班级名称	50		
Grade	数字	年级			
SchoolID	数字	所属学校编号			
CreateDate	日期/时间	创建时间		Now()	
AdminUser	数字	班级管理员所对应的 ID			
SHTag	文本	班级管理员的状态标志	1	'0'	

说明： 字段 CreateDate 的默认值为系统函数 Now()，该函数用于获取系统的当前时间。字段 SchoolID 表示班级所属的学校编号，该字段与 SchoolInfo 数据表中的字段 ID 相关联。字段 AdminUser 表示管理员所对应的用户编号，该字段与 UserSheet 数据表中的字段 Code 相关联。字段 SHTag 表示管理员的状态标志，其值为 0 表示尚无管理员，为 1 表示已有人申请管理员但未经过半数以上班级成员的同意，为 2 表示已有管理员且申请成功。

2. 班级留言信息表(LYInfo)

LYInfo 数据表主要用于存储班级留言的基本信息，其表结构见表 10.2。

表 10.2　LYInfo 数据表

字 段 名	数据类型	字段描述	字段大小	默 认 值	主　键
TheNo	自动编号	留言编号	长整型		P
UserID	数字	发表留言的用户编号			
ClassID	数字	对应的班级编号			
Title	文本	留言标题	100		
Content	文本	留言内容	200		
TTime	日期/时间	留言时间		Now()	

说明：字段 UserID 表示发表留言的用户编号，该字段与 UserSheet 数据表中的字段 Code 相关联。字段 ClassID 表示发表留言对应的班级编号，该字段与 ClassInfo 数据表中的字段 ID 相关联。

3. 班级相片信息表(PhotoInfo)

PhotoInfo 数据表主要用于存储班级相片的基本信息，其表结构见表 10.3。

表 10.3　PhotoInfo 数据表

字 段 名	数据类型	字段描述	字段大小	默 认 值	主　键
ID	自动编号	相片编号	长整型		P
UpUser	数字	上传用户编号			
ClassCode	数字	相片对应的班级编号			
PhotoAddr	文本	相片地址	100		
PhotoName	文本	相片名称	100		
ClickNum	数字	点击次数		0	
UpDate	日期/时间	相片上传时间		Now()	

说明：字段 UpUser 表示上传相片的用户编号，该字段与 UserSheet 数据表中的字段 Code 相关联。字段 ClassCode 表示上传相片所对应的班级编号，该字段与 ClassInfo 数据表中的字段 ID 相关联。字段 PhotoAddr 表示相片地址，这里仅存储相片文件的文件名，其文件路径默认为系统目录下的 Photo 子目录。

4. 相片评论信息表(Photo_LY)

Photo_LY 数据表主要用于存储班级相片的评论信息，其表结构见表 10.4。

5. 学校基本信息表(SchoolInfo)

SchoolInfo 数据表主要用于存储学校的基本信息，其表结构见表 10.5。

表 10.4　Photo_PL 数据表

字 段 名	数据类型	字段描述	字段大小	默 认 值	主　键
ID	自动编号	相片评论编号	长整型		P
PhotoID	数字	相片编号			
Pl_User	数字	发表评论的用户编号			
Pl_Date	日期/时间	评论时间		Now()	
Pl_Content	备注	评论内容			

说明： 字段 PhotoID 表示评论所针对的相片编号，该字段与 PhotoInfo 数据表中的字段 ID 相关联。字段 Pl_User 表示发表评论的用户编号，该字段与 UserSheet 数据表中的字段 Code 相关联。

表 10.5　SchoolInfo 数据表

字 段 名	数据类型	字段描述	字段大小	默 认 值	主　键
ID	自动编号	学校编号	长整型		P
SchoolName	文本	学校名称	50		
Province	文本	所属省份	50		
Type	文本	学校类型	50		
CreateDate	日期/时间	创建时间		Now()	

6. 省份参数信息表(Sys_Province)

Sys_Province 数据表主要用于存储各省份的相关参数信息，其表结构见表 10.6。

表 10.6　Sys_Province 数据表

字 段 名	数据类型	字段描述	字段大小	默 认 值	主　键
ID	自动编号	省份编号	长整型		P
Province	文本	省份名称	50		

7. 用户班级关联信息表(UserClass)

UserClass 数据表主要用于存储用户与班级的关联信息，其表结构见表 10.7。

表 10.7　UserClass 数据表

字 段 名	数据类型	字段描述	字段大小	默 认 值	主　键
UserCode	数字	用户编号			P
ClassCode	数字	班级编号			S

续表

字 段 名	数据类型	字段描述	字段大小	默 认 值	主　键
InTime	日期/时间	加入时间		Now()	
HF_Admin	文本	批复班级申请标志	1	'0'	

说明：字段 UserCode 表示关联的用户编号，该字段与 UserSheet 数据表中的字段 Code 相关联。字段 ClassCode 表示关联的班级编号，该字段与 ClassInfo 数据表中的字段 ID 相关联。在本表中，字段 UserCode 和字段 ClassCode 合为主键。

8. 用户基本信息表(UserSheet)

UserSheet 数据表主要用于存储用户的基本信息，其表结构见表 10.8。

表 10.8　UserSheet 数据表

字 段 名	数据类型	字段描述	字段大小	默 认 值	主　键
Code	自动编号	用户编号	长整型		
LoginName	文本	用户的登录名称	50		P
UserPass	文本	用户的登录密码	50		
UserName	文本	用户的真实名称	50		
Birth	日期/时间	出生日期			
Email	文本	邮箱地址	100		
PassQuestion	文本	密码提示问题	100		
PassAnswer	文本	密码提示问题的答案	100		
Address	文本	联系地址	100		
Tel1	文本	联系的固定电话	50		
Tel2	文本	联系的移动电话	50		
Works	文本	用户的工作单位	100		
ZipCode	文本	邮政编码	50		
Sex	文本	用户性别	50		
JHTag	文本	账号是否激活标识	1	'0'	
LastLogin	日期/时间	上次登录时间			
CurrLogin	日期/时间	本次登录时间			
Nums	数字	登录次数		0	

10.1.3　站点设置

本例中，站点设置的本地信息如图 10.1 所示。

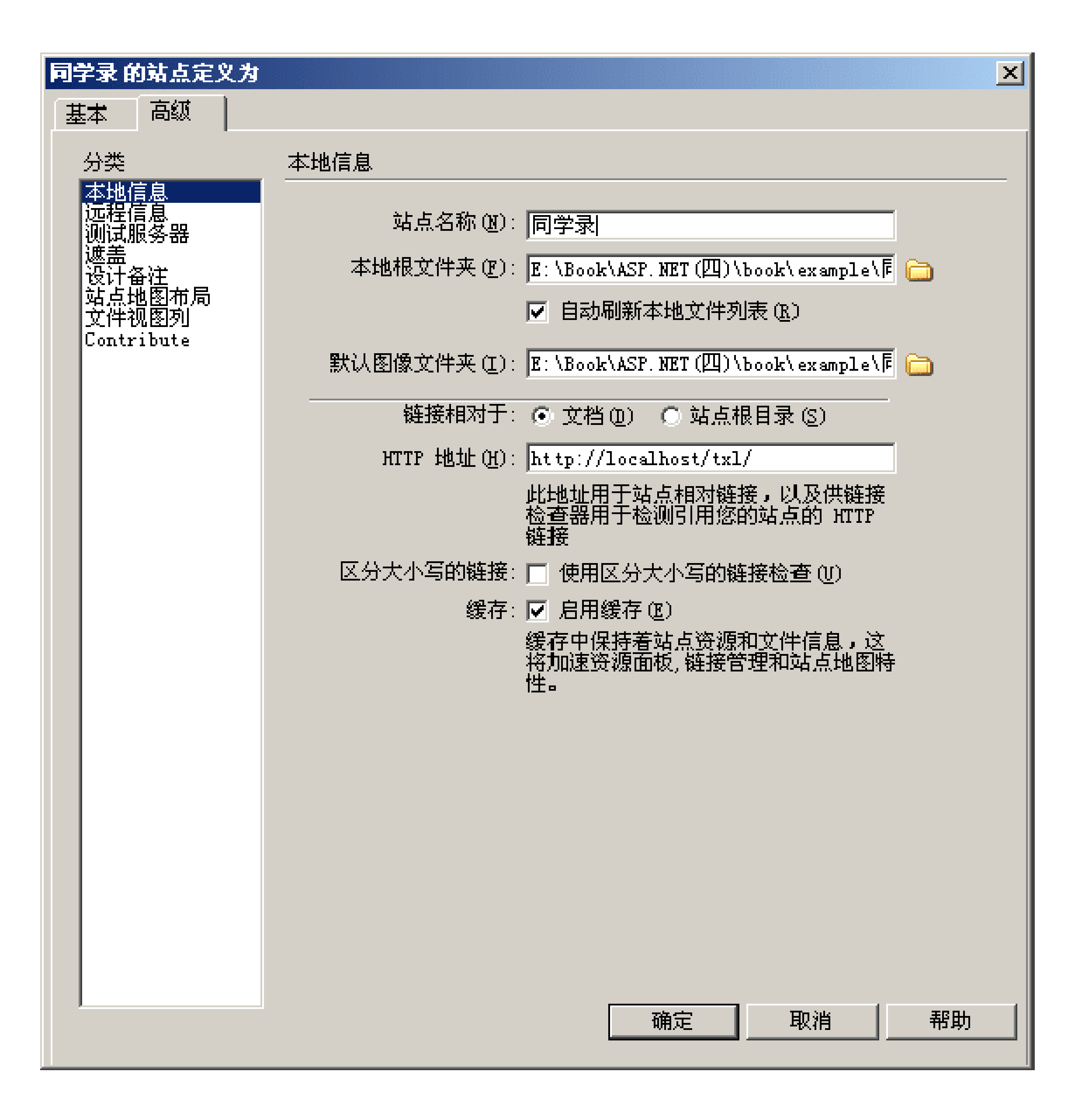


图 10.1　本地信息

其中，站点名称设置为“同学录”，本地根文件夹设置为同学录系统所在的磁盘目录，默认图像文件夹则设置为同学录系统下的 Image 目录，HTTP 地址设置为“http://localhost/txl”(在此之前，需将网络相册系统设置为 Web 共享，并设置其共享名为 txl)。

站点设置中，远程信息设置如图 10.2 所示。

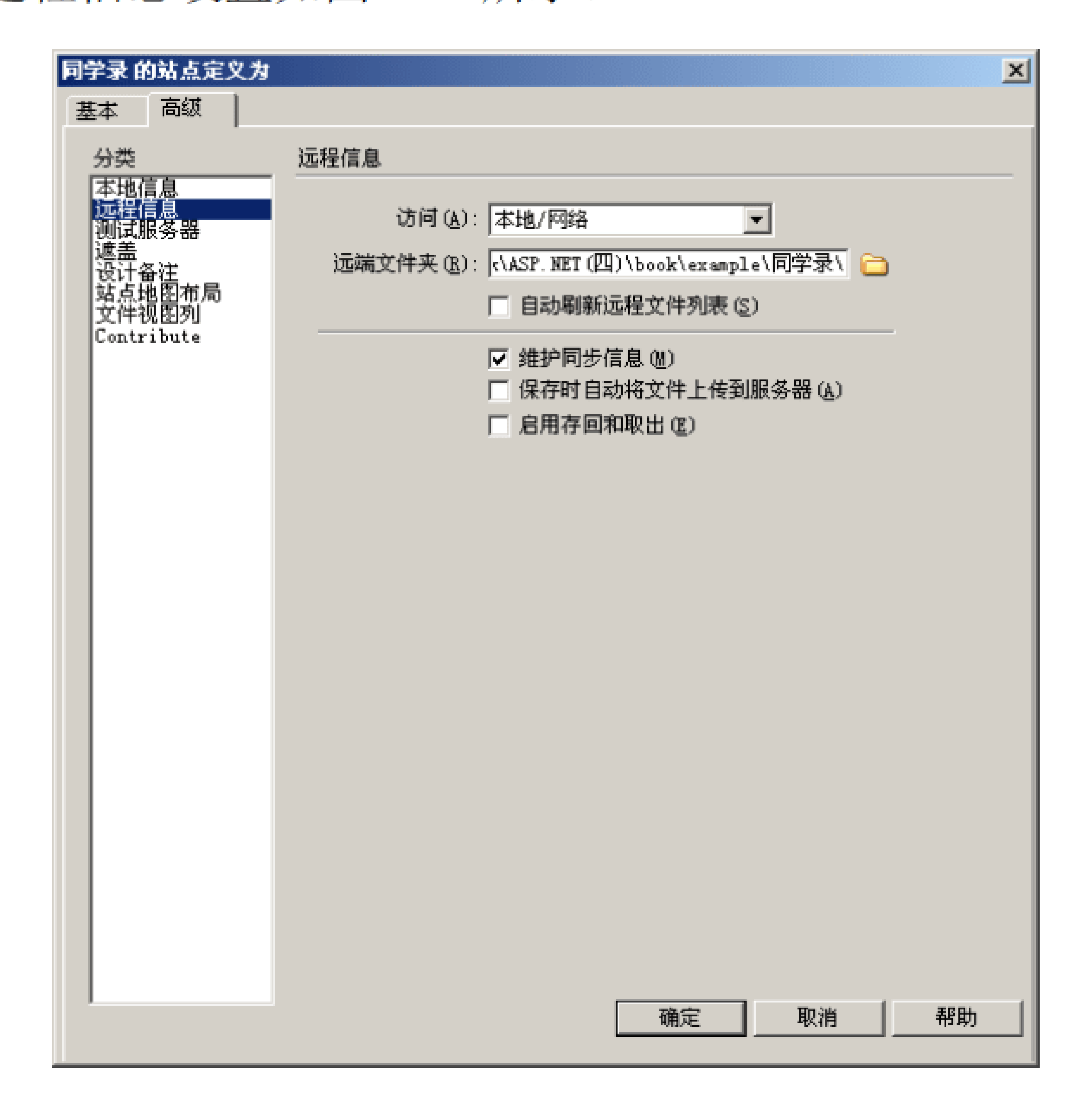


图 10.2　远程信息

由于对同学录系统的设计、测试和运行均是在本机上操作的，因此这里将访问方式设置为【本地/网络】，而远端文件夹则与本地根文件夹设置为相同的目录。

站点设置时，测试服务器设置如图 10.3 所示。

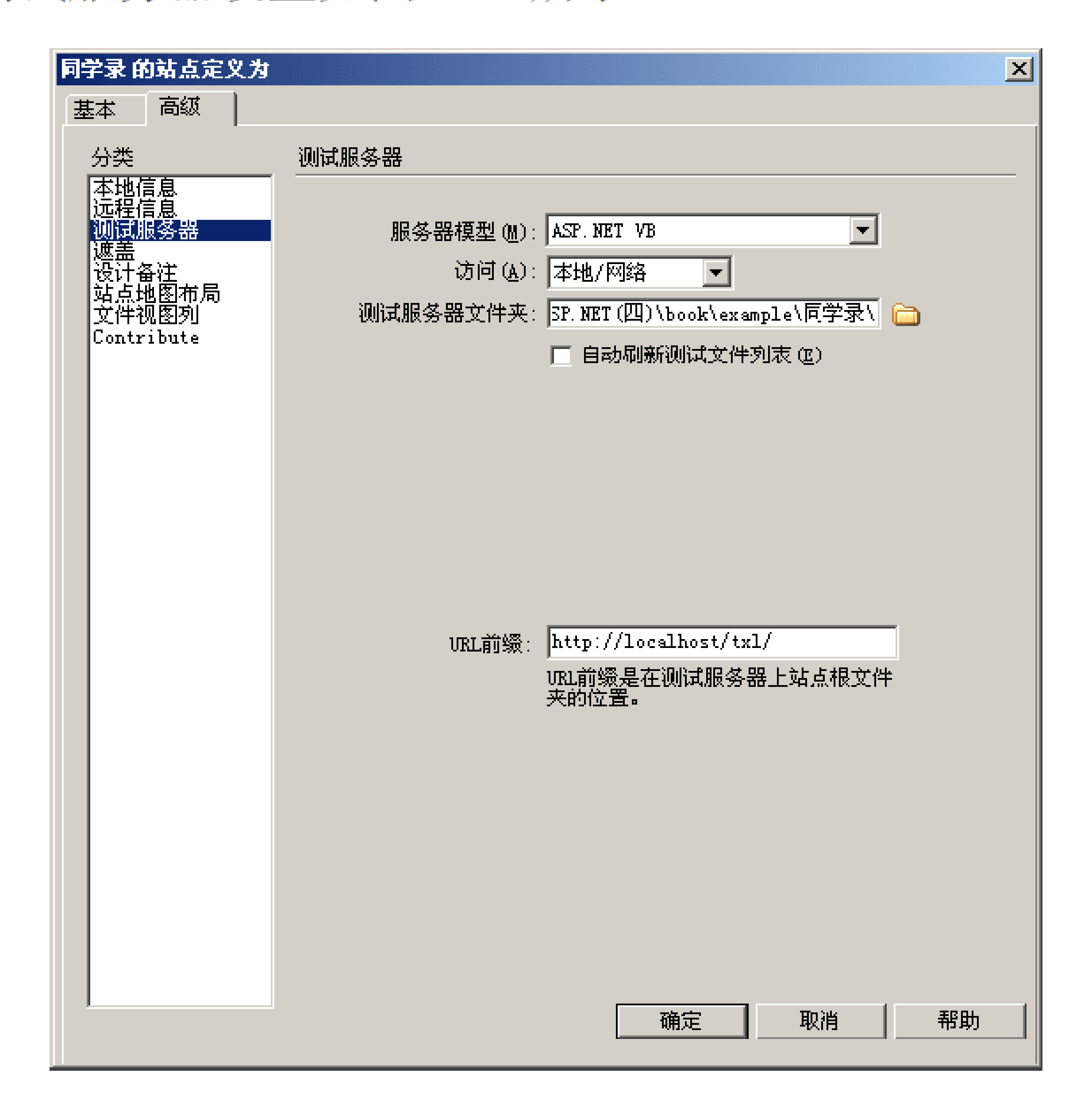


图 10.3　测试服务器

对于站点的部署，请参考本书第 7 章，这里不再赘述。

10.2　构 建 框 架

10.2.1　创建模板

在实现同学录的各项功能之前，先创建一个基本的页面模板。

(1)　选择【文件】|【新建】命令，在弹出的【新建文档】对话框中，选择【模板页】|【ASP.NET VB 模板】，如图 10.4 所示。单击【创建】按钮，新建一个模板页。

图 10.4　【新建文档】对话框

(2)　导入外部样式表。打开 CSS 面板，右击，在弹出的快捷菜单中选择【附加样式表】命令，此时将弹出【链接外部样式表】对话框，如图 10.5 所示。

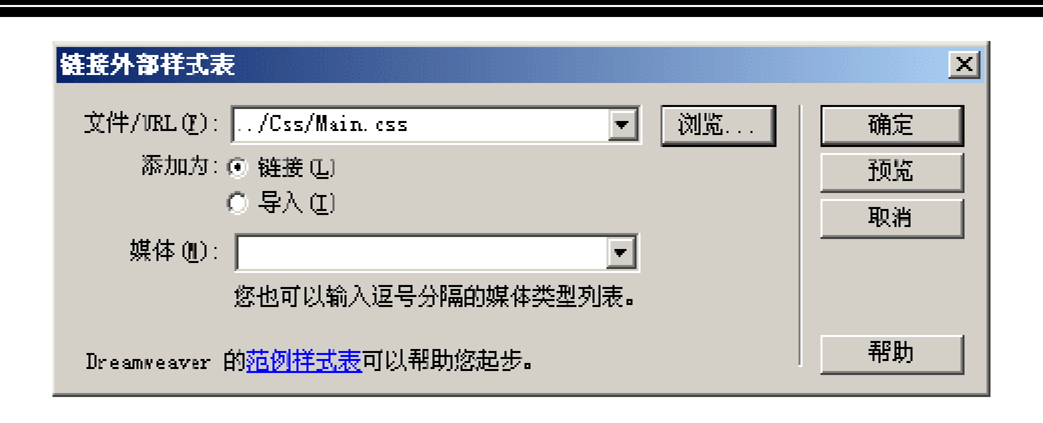


图 10.5　【链接外部样式表】对话框

(3) 单击【浏览】按钮，选择系统目录中的 CSS 文件夹下的 Main.css 文件，设置添加方式为【链接】。单击【确定】按钮，完成样式表的附加。

(4) 切换至【设计】视图并右击，从弹出的快捷菜单中选择【页面属性】命令。在弹出的【页面属性】对话框的【外观】分类下，设置页面的上边距为 0，如图 10.6 所示。单击【确定】按钮后，就完成了页面设置。

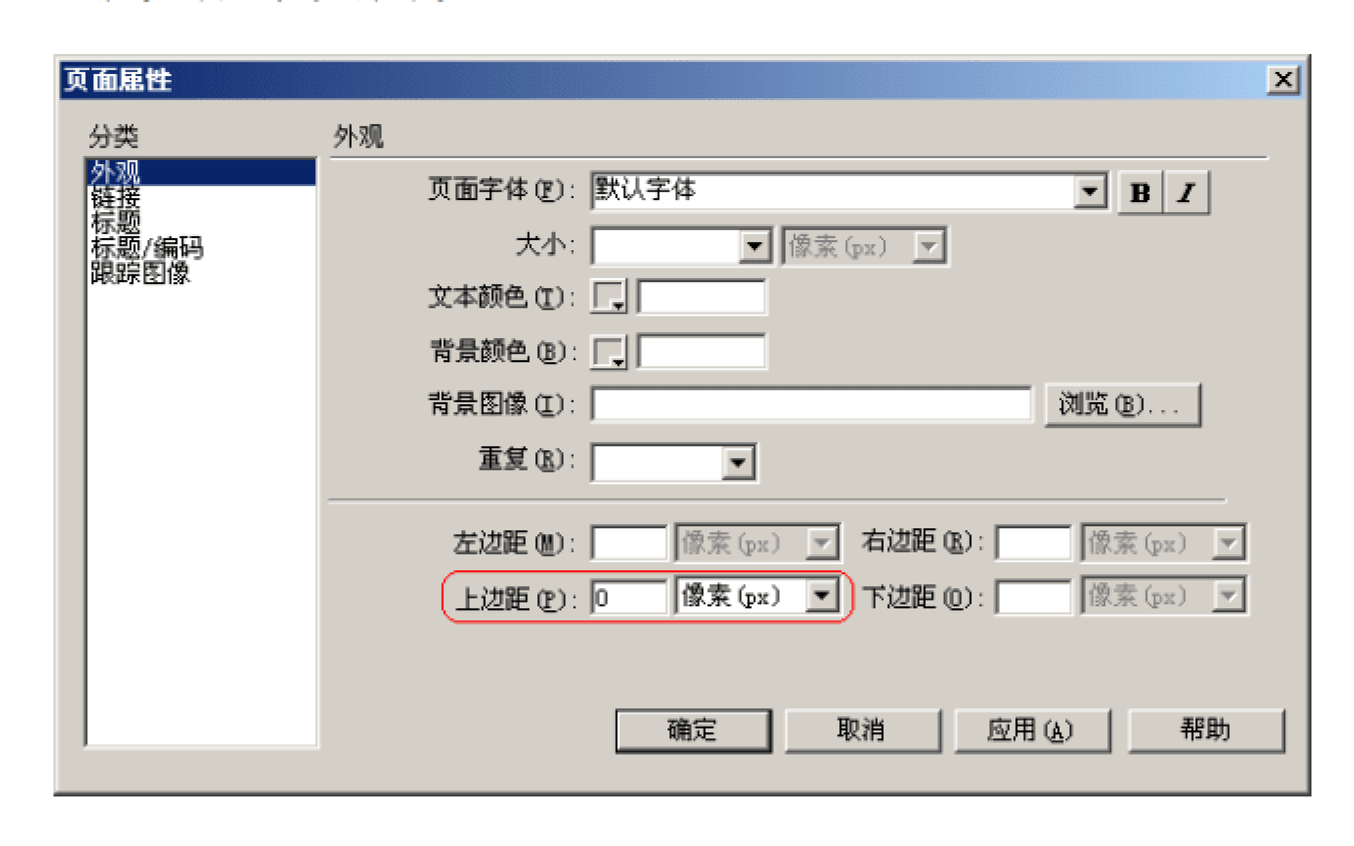


图 10.6　【页面属性】对话框

(5) 选择【插入】|【表格】命令，在页面上插入一个 2 行 1 列的表格 table1，其属性设置如图 10.7 所示。

图 10.7　表格属性

(6) 在表格 table1 的第 1 行中，插入一个图片，其源文件设置为 Image 文件夹下的 Title.gif；将光标置于第 2 行中，选择【插入】|【模板对象】|【可编辑区域】命令，此时将弹出【新建可编辑区域】对话框，如图 10.8 所示。

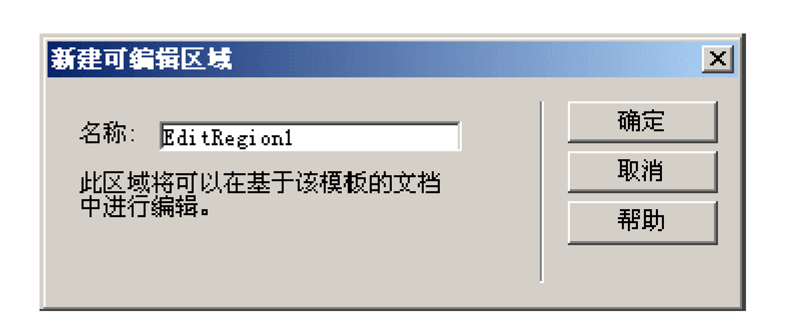


图 10.8　【新建可编辑区域】对话框

(7) 设置名称为 EditRegion1，单击【确定】按钮，完成可编辑区域的设置。同时，在可编辑区域中插入一个表单 Form，并设置其 Runat 属性为 Server，此时的页面如图 10.9 所示。

图 10.9　页面设计

(8) 选择【文件】|【保存】命令，将模板命名为“页面模板”，并加以保存，如图 10.10 所示。此时，系统会自动在同学录的系统目录下新建一个名为 Templates 的文件夹，并在其中生成一个名为“页面模板.dwt.aspx”的页面文件，这就是刚才所创建的模板。

图 10.10　保存模板

至此，模板创建完成，可编辑区域 EditRegion1 即为在其他页面添加相关功能的区域。

10.2.2　用户登录

用户登录是用户进入同学录系统的首页面，其主要功能是验证用户的合法身份。

(1) 新建一个 ASP.NET VB 类型的页面，将其命名为 Default.aspx。选择【修改】|【模板】|【套用模板到页】命令，在弹出的【选择模板】对话框中，选择前面创建的模板“页面模板”，如图 10.11 所示。

图 10.11　选择模板

单击【确定】按钮，将模板应用到本页面，并将页面标题设置为“用户登录”。

(2) 在可编辑区域 EditRegion1 中的表单 Form 中插入一个 5 行 2 列的表格，表格属性

的设置如图 10.12 所示。

图 10.12　表格属性

(3) 在表格中，插入相应的文本、文本框控件、按钮控件以及 RequiredFieldValidator 验证控件等。这里，采用了与第 8 章中的会员管理类似的登录页面，如图 10.13 所示。

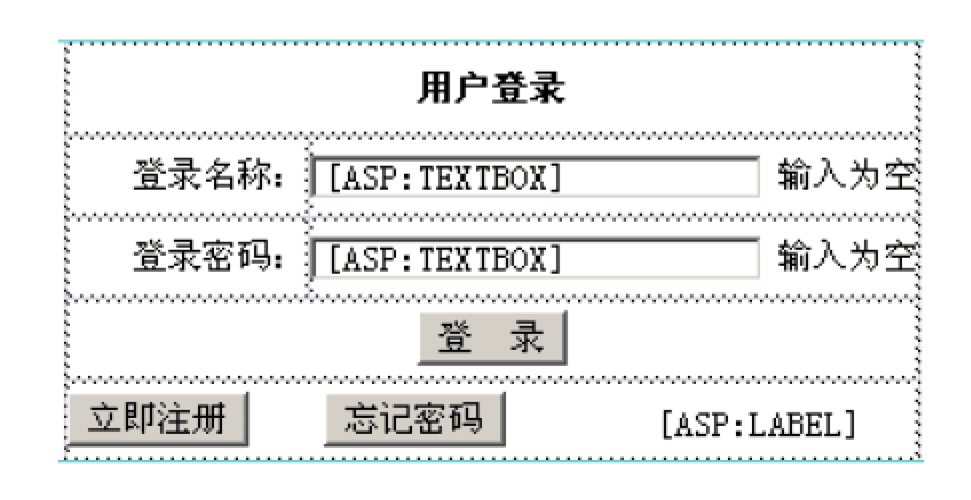


图 10.13　界面设计

对于表格中的各种控件元素的定义，读者可参考 8.3.1 小节。

为了验证用户输入的登录名称和登录密码的正确性，下面将创建一个数据集。

(4) 打开【服务器行为】面板，单击按钮，从弹出的菜单中选择【数据集】命令，将弹出【数据集】对话框，如图 10.14 所示。

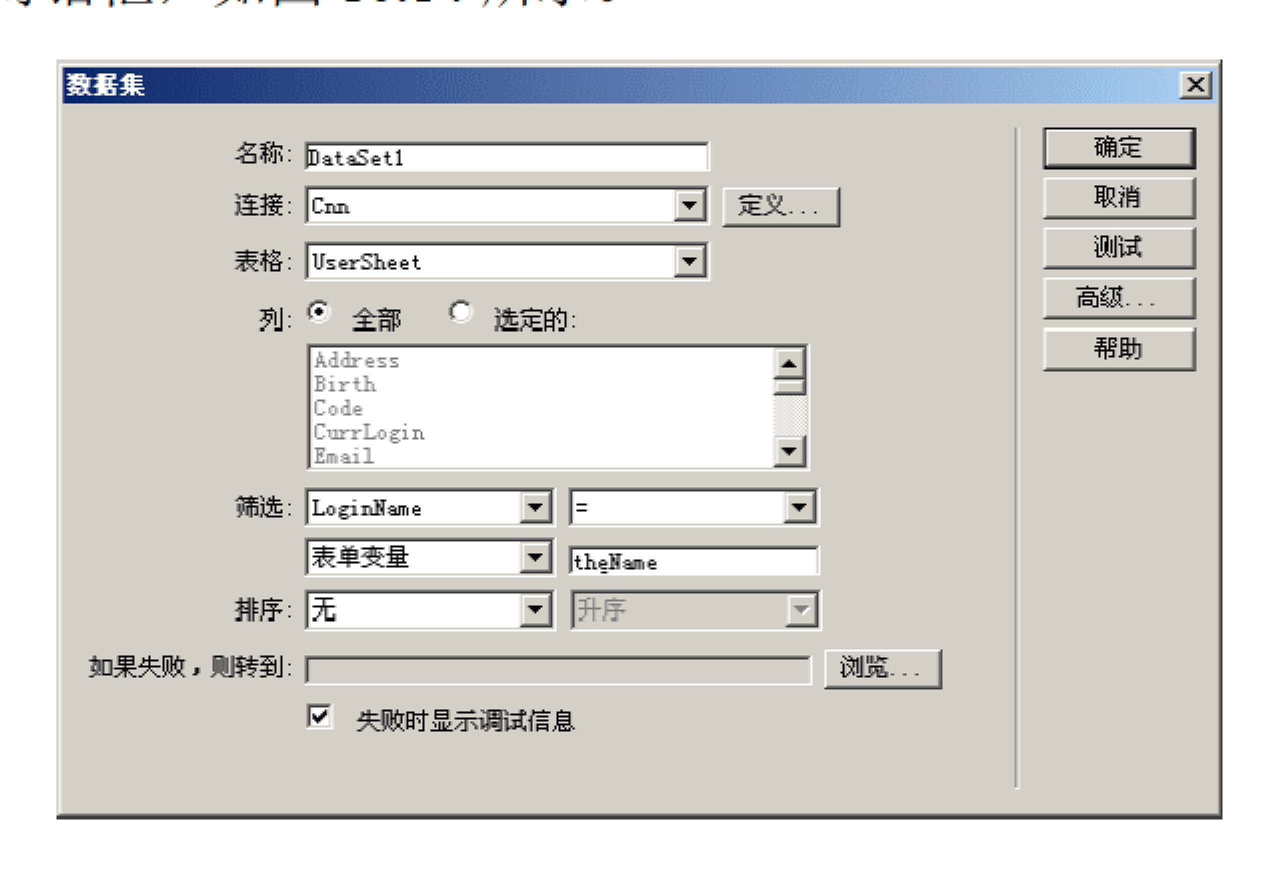


图 10.14　【数据集】对话框

设置数据集名称为 DataSet1，数据库连接选择 Cnn，在【表格】下拉列表框中选择 UserSheet，设置筛选条件为字段 LoginName 等于表单变量 theName。单击【高级】按钮，将对话框切换至高级模式，以便添加一个筛选条件，判断密码的正确性。

(5) 在【数据集】对话框的高级模式中，单击【添加参数】按钮，添加一个新的参数，并设置参数的名称为“@Pass”，其类型为“VarChar”，其参数值为表单变量 thePwd 的值。然后，在 SQL 文本框中的 WHERE 子句之后添加一个条件表达式，增加对登录密码的判断，如图 10.15 所示。

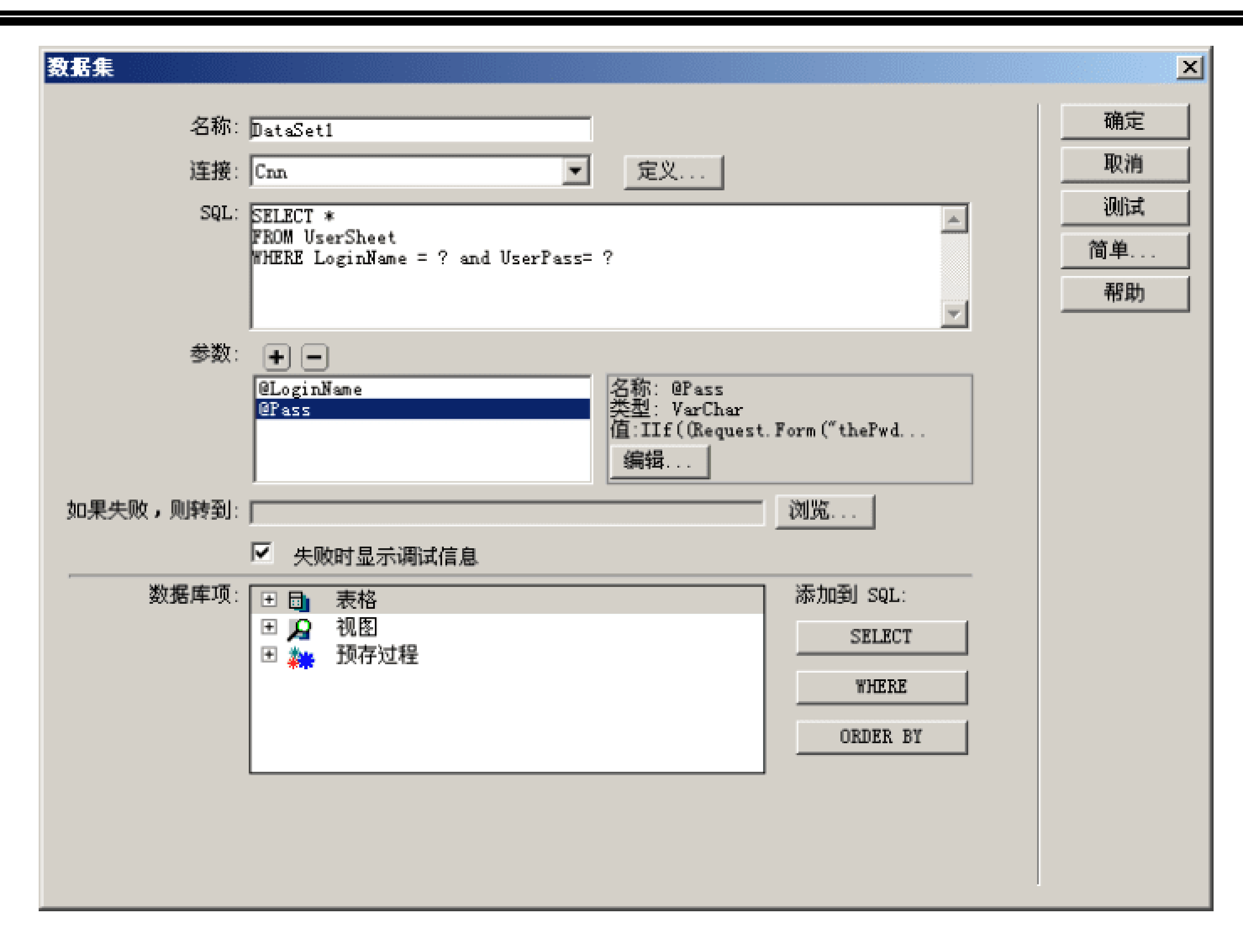


图 10.15　【数据集】对话框

(6)　单击【确定】按钮，完成数据集的创建。当用户输入登录名称和登录密码并单击【登录】按钮时，根据数据集 DataSet1 中的记录是否为空，可以判断用户所输入的用户名及密码是否正确。

(7)　将视图切换至【代码】视图，添加各个按钮的单击事件。

对于【登录】按钮，其所执行的操作是首先根据数据集 DataSet1 的记录是否为空判断用户所输入的登录名称和密码是否正确，然后根据数据集 DataSet1 中的字段 JHTag 的值是否为 1 来判断用户的账号是否已被激活。如果账号已被激活，则表示用户可正常登录，此时将用户 ID、用户名称及当前登录时间分别赋予 3 个 Cookie 变量，同时更新用户信息表中的上次登录时间、本次登录时间和登录次数等信息，并将页面跳转至登录后的主页面 Main.aspx。

【登录】按钮的事件代码如下：

【示例代码】

```
Sub Login_Click(ByVal sender As Object, ByVal E As EventArgs)
  Dim JHStr as String
  Dim StrCnn As String
  Dim StrSql As String
  Dim Cnn As OleDbConnection
  Dim Cmd As OleDbCommand
  '判断数据集的记录是否大于 0
  If DataSet1.DefaultView.Table.Rows.Count>0 then
    '数据集的记录大于 0，用户所输入的名称及密码正确
    '获取当前用户账号的激活标志
    JHStr=DataSet1.DefaultView.Table.Rows(0)("JHTag")
    If trim(JHStr)="1" Then
      '激活标志为 1，账号已激活
      Response.Cookies("id") = DataSet1.DefaultView.Table.Rows(0)("code")
'将用户编码赋予 Cookies 变量 ID
      Response.Cookies("name") =
DataSet1.DefaultView.Table.Rows(0)("username")  '将用户名称赋予 Cookies 变量
```

```
Name
      '将当前登录时间赋予 Cookies 变量 LoginTime
      Response.Cookies("logintime") = System.DateTime.Now
      '获取数据库连接字符串，并创建数据库连接
      StrCnn =
System.Configuration.ConfigurationSettings.AppSettings("MM_CONNECTION_ST
RING_Cnn")
      Cnn = New OleDbConnection(StrCnn)
      Cnn.Open()
      '更新用户信息表中的上次登录时间、本次登录时间以及登录次数信息
      StrSql = "update usersheet set lastlogin=currlogin,currlogin='" &
System.DateTime.Now & "',nums=nums+1 where code=" &
DataSet1.DefaultView.Table.Rows(0)("code")
      Cmd = New OleDbCommand(StrSql, Cnn)
      Cmd.ExecuteNonQuery()
      Cnn.Close()  '关闭数据库连接
      '将页面跳转至登录后的主页面
      Response.Redirect("main.aspx")
    else
      '账号未激活，提示相关信息
      LblErr.Text = "此账户尚未激活，无法正常登录！"
    end if
  else
    '数据集 DataSet1 的记录为空，提示错误信息
    LblErr.Text = "登录名称或密码错误！"
  end if
End Sub
```

对于【立即注册】按钮和【忘记密码】按钮，所执行的操作均是将页面直接跳转至相应的功能页面，其代码如下：

【示例代码】

```
'【忘记密码】按钮的单击事件
Sub Pass_Click(ByVal sender As Object, ByVal E As EventArgs)
  Response.Redirect("LoadPass.aspx")
End Sub
'【立即注册】按钮的单击事件
Sub Reg_Click(ByVal sender As Object, ByVal E As EventArgs)
  Response.Redirect("Login.aspx")
End Sub
```

为了简化系统，这里的用户注册功能和找回密码功能均沿用了第 8 章会员管理系统中对应功能的相关页面 Login.aspx 和 LoadPass.aspx。对于这两个页面的功能实现，这里不再介绍，读者可直接参考 8.2 节和 8.4 节。

提示： 本页面需导入 System.Data 和 System.Data.OleDb 两个命名空间。只要在页面的代码中存在对数据库的操作(即不是通过【服务器行为】面板来对数据库进行操作)，均需在页面中导入以上两个命名空间。对于这一点，在后面的章节中将不再提示。

至此，用户登录页面设计完成，页面预览如图 10.16 所示。

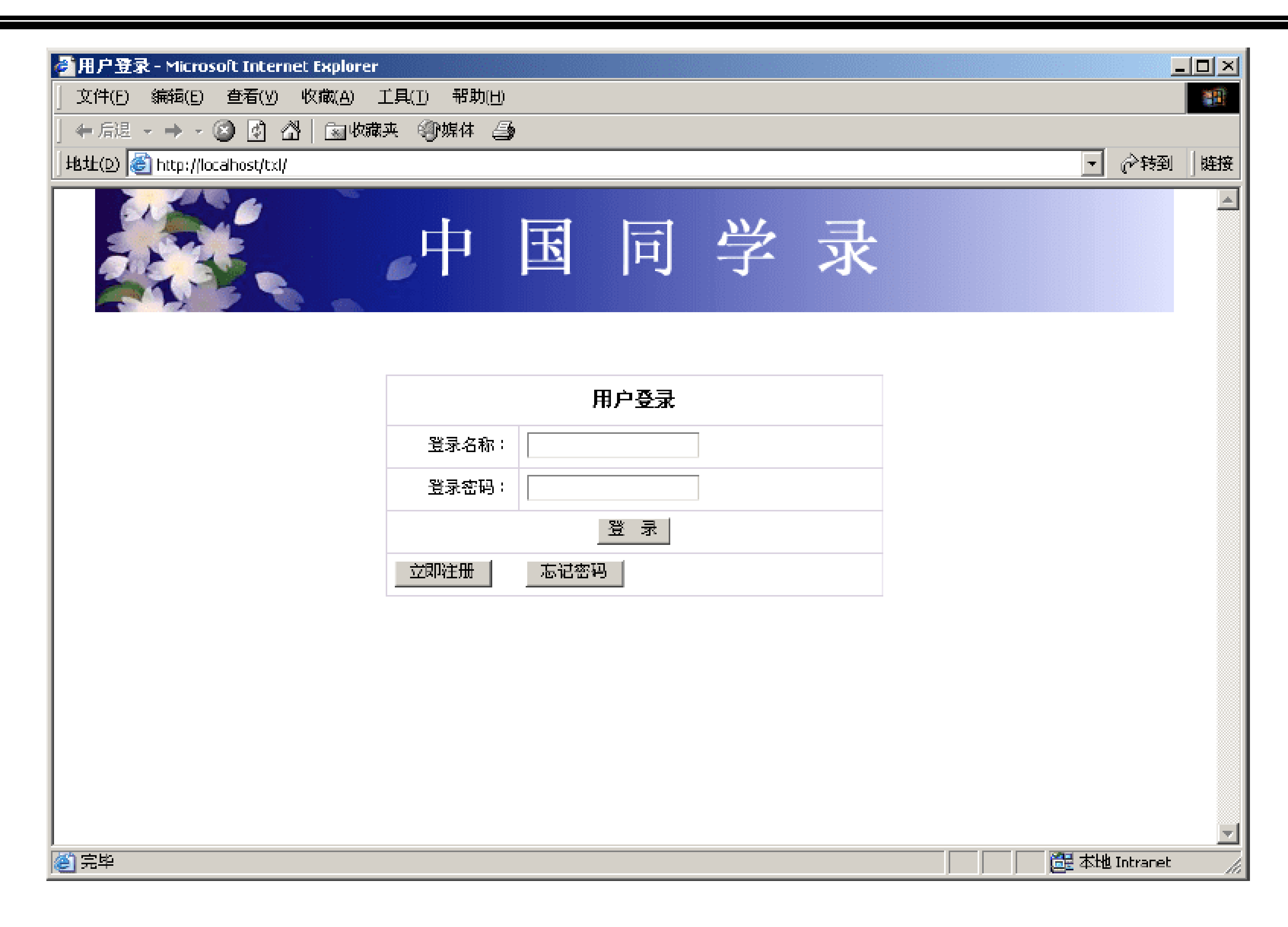


图 10.16　页面预览

10.3　系 统 主 页

10.3.1　主页框架

通过在登录界面输入正确的用户名和密码后，即可进入同学录系统的主页。在系统主页中，可分为三大块：个人信息、我的同学录和班级信息。接下来，介绍框架的建立及功能区域的划分。

(1)　新建一个 ASP.NET VB 类型的页面，将其命名为 Main.aspx。选择【修改】|【模板】|【套用模板到页】命令，在弹出的【选择模板】对话框中，选择【页面模板】。单击【确定】按钮，将模板应用到本页面。

(2)　在可编辑区域 EditRegion1 中的表单 Form 中，插入一个 1 行 2 列的表格 Table1，表格属性按如图 10.17 所示进行设置。

图 10.17　表格属性

(3)　在第 1 个单元格中，将显示个人信息及我的同学录，其中，我的同学录又包括我的班级和我的学校两部分；而在第 2 个单元格中，将主要显示班级信息，包括班级基本信息、最近到访的同学和班级最新相片等。

页面的功能区域的划分如图 10.18 所示。

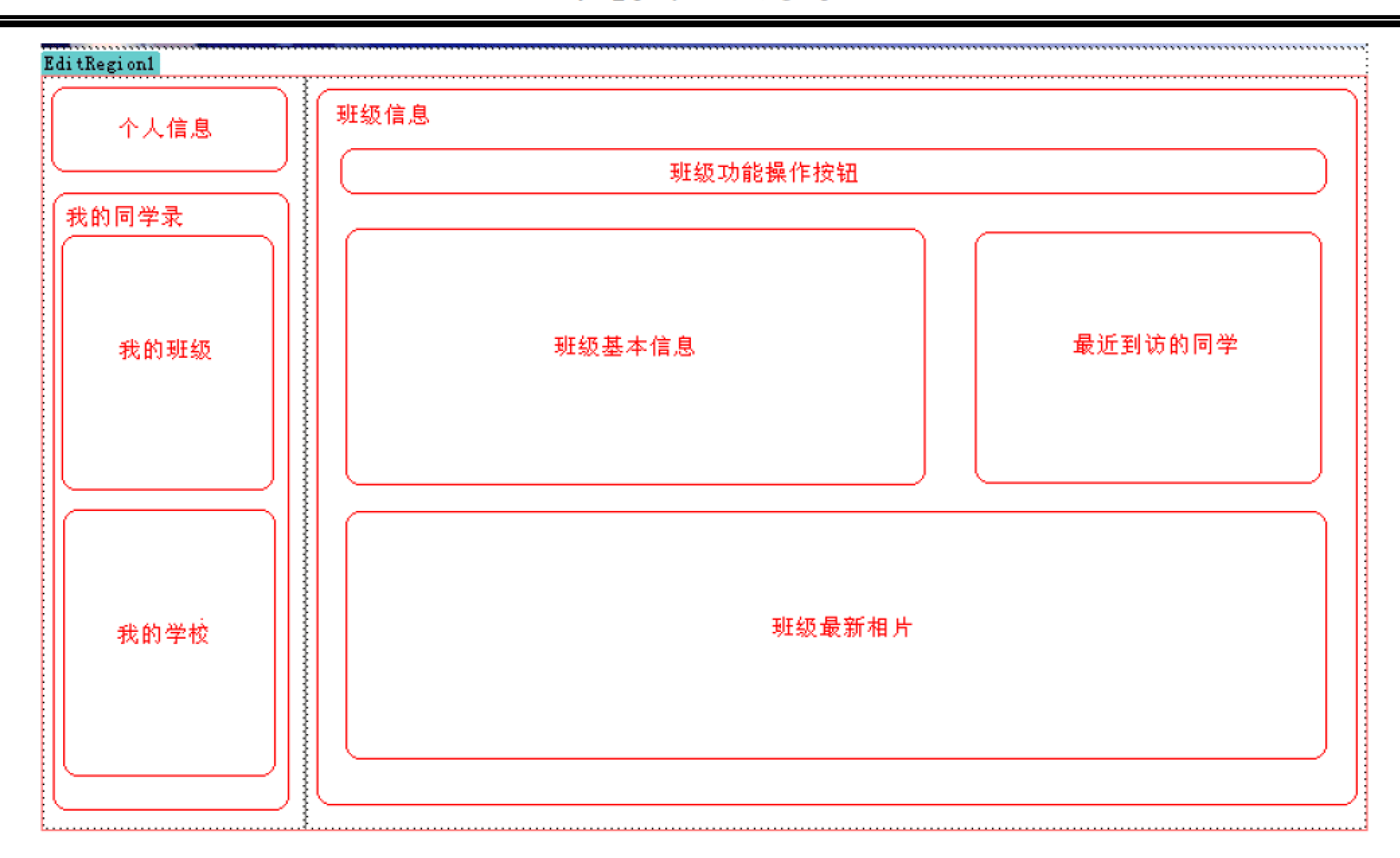


图 10.18　页面功能区域

下面逐步介绍如何实现各区域所对应的功能。

10.3.2　个人信息

这里的个人信息并非指当前登录用户的详细资料，而是指当前用户的登录信息，如用户的上次登录时间以及登录次数等。

(1)　在表格 table1 的第 1 个单元格中，插入一个 3 行 3 列的表格 table2，其属性设置如图 10.19 所示。

图 10.19　表格属性

(2)　合并表格 table2 第 1 行的 3 个单元格，并在【属性】面板中设置合并单元格的背景图片为 image/bg_1.gif；合并表格 table2 第 3 行中的 3 个单元格，并在【属性】面板中设置合并单元格的背景图片为 image/bg_4.gif。

接着，分别设置第 2 行的第 1 个单元格和第 3 个单元格的背景图片分别为 image/bg_2.gif 和 image/bg_3.gif。

(3)　在表格 table2 的第 2 行的单元格中插入一个 5 行 1 列的表格 table3，其属性设置如图 10.20 所示。

事实上，表格 table3 才是真正用于控制显示个人信息的表格，表格 table2 仅用来实现边框效果。

(4)　在表格 table3 的各单元格中输入相应的文本，并插入一个 LinkButton 控件，如图 10.21 所示。

图 10.20　表格属性

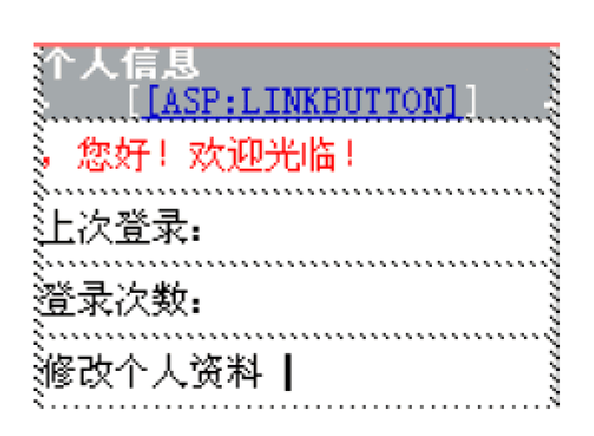


图 10.21　设置表格

表格中所添加的 LinkButton 控件，主要用于实现安全退出的功能。在其属性设置中，ID 值为 LinkButton1，前景色为白色，OnClick 事件为 Exit_Click，如图 10.22 所示。

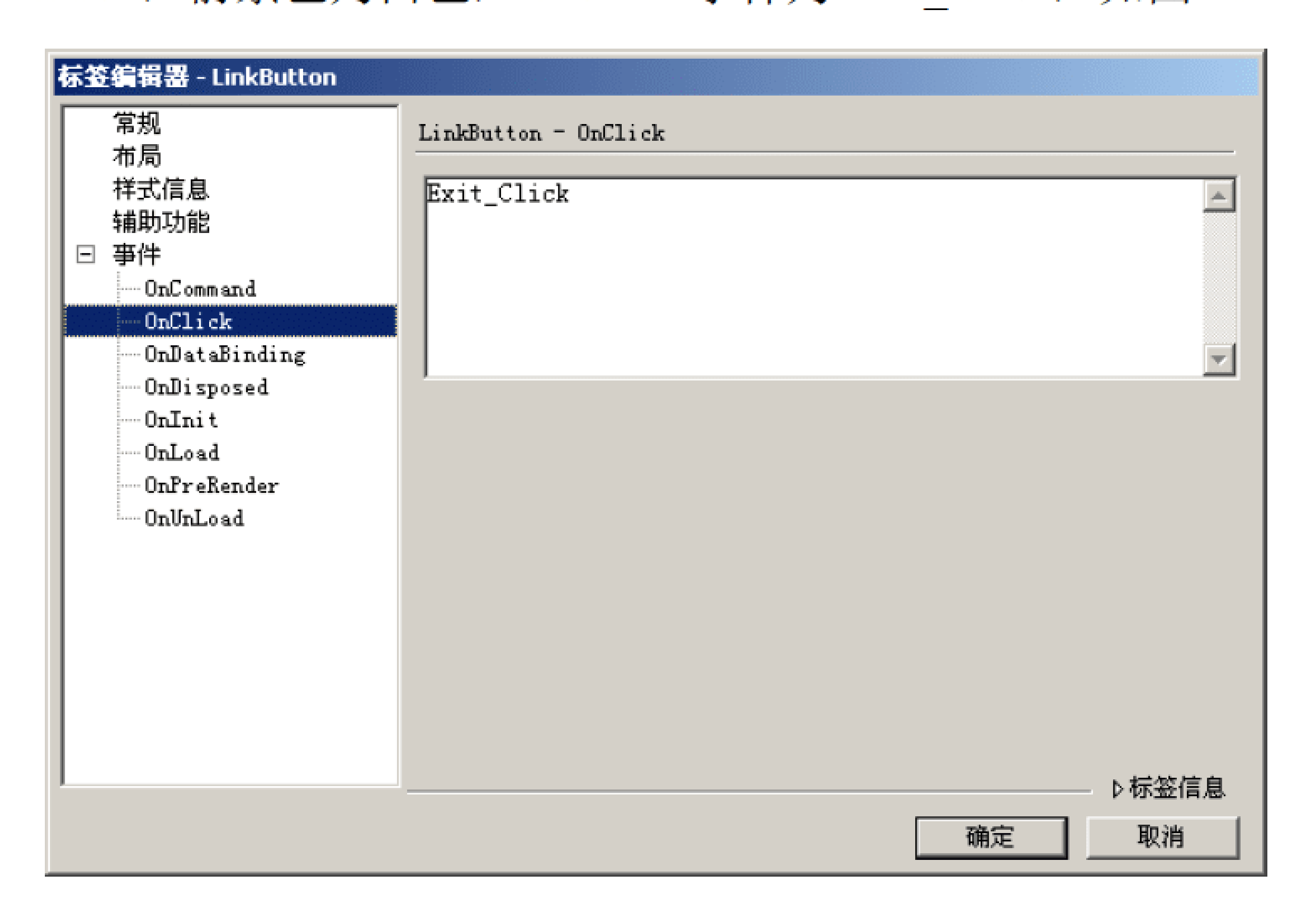


图 10.22　LinkButton 控件的单击事件

接下来，将创建一个数据集来绑定个人信息中相关数据的显示。

(5)　在【应用程序】面板组中，切换至【服务器行为】面板，单击 + 按钮，在弹出的菜单中选择【数据集】命令，在弹出的【数据集】对话框中，设置数据集名称为 DataSet1，连接选择 Cnn，表格选择 UserInfo，设置筛选条件为字段 Code 等于 Cookie 变量 ID，如图 10.23 所示。

单击【确定】按钮，完成数据集的创建。

(6)　在【应用程序】面板组中，将选项切换至【绑定】面板，选择并展开数据集 DataSet1，如图 10.24 所示。

(7)　选择并拖动字段 UserName 至表格 table3 中的文本“，您好”之前，选择并拖动字段 LastLogin 至文本“上次登录：”之后，选择并拖动字段 Nums 至文本“登录次数：”之后。然后，选择文本“修改个人资料”，并在【属性】面板中设置其链接为 ModiUser.aspx。

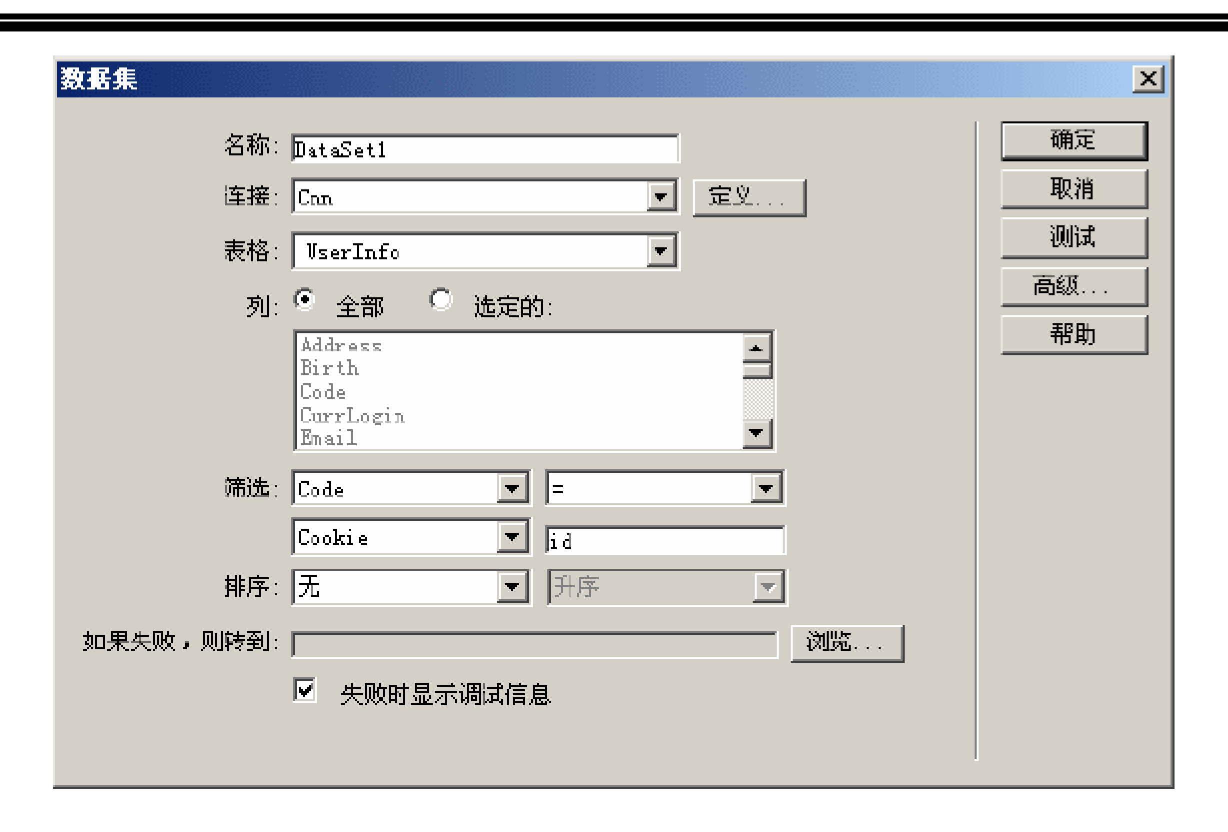


图 10.23　创建数据集 DataSet1

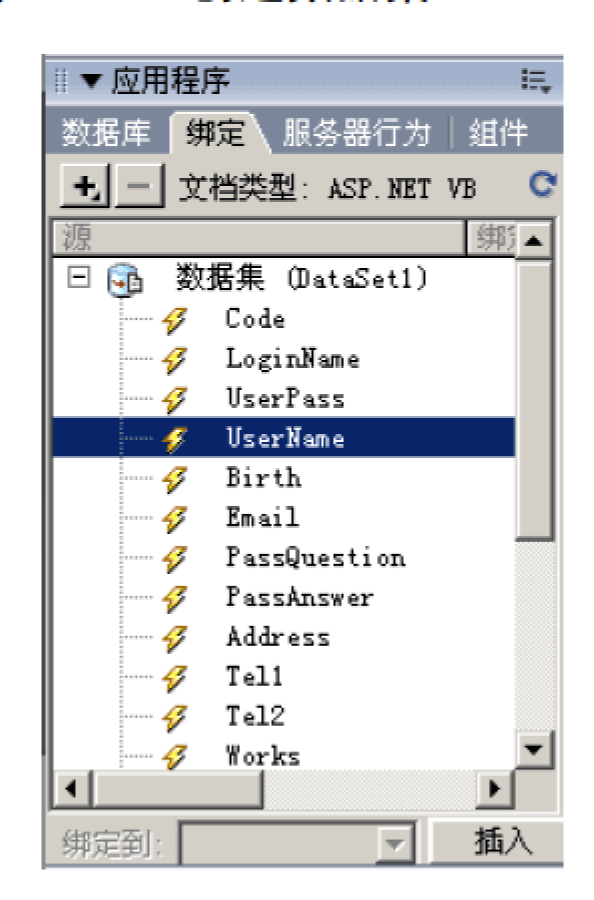


图 10.24　【绑定】面板

切换至【代码】视图，添加【退出】链接的单击事件，其代码如下：

【示例代码】

```
'【退出】的单击事件
Sub Exit_Click(ByVal Sender As Object, ByVal E As EventArgs)
  '清空 Cookies 变量
  Response.Cookies("id").Value = ""
  Response.Cookies("name").Value = ""
  Response.Cookies("logintime").Value = ""
  '将页面跳转至登录页面
  Response.Redirect("default.aspx")
End Sub
```

至此，个人信息的显示设计完成，此部分的预览效果如图 10.25 所示。

图 10.25　“个人信息”部分的预览效果

此外，由于本系统的用户信息与会员管理中的用户信息采用了相似的数据表结构，因此这里的“修改个人资料”功能仍将沿用会员管理系统中的修改个人信息页面 ModiUser.aspx。但由于会员管理系统中对登录用户的标识是采用 Session 变量，而本系统中是采用 Cookies 变量，因此需要对原页面稍做修改，操作如下。

(1) 将“会员管理”系统中的 ModiUser.aspx 页面复制到“同学录”系统目录下。

(2) 打开该文件，切换至【代码】视图。在此文件中，有两处调用了 Session 变量 userid 来获取当前的登录用户信息，一处是使用【更新记录】命令设置的参数定义，另一处是在 Page_Load 事件中，如图 10.26 所示。

```
DatabaseType='<%# System.Configuration.ConfigurationSettings.AppSettings("MM_CONNECTION_DATABASETYPE_Cnn")
 %>'
CommandText='<%# "SELECT * FROM UserSheet WHERE Code = ?" %>'
Debug="true"
>
  <Parameters>
    <Parameter  Name="@Code"  Value='<%# IIf((Not Session("userid") Is Nothing), Session("userid"), "") %>'
'  Type="Integer"   />
  </Parameters>
</MM:DataSet>
<MM:PageBind runat="server" PostBackBind="false" />
<script runat="server">
  Sub Page_Load(ByVal Sender As Object, ByVal E As EventArgs)
    If (Not Page.IsPostBack) Then
      If IsDBNull(Session("userid")) Or Trim(Session("userid")) = "" Then
        Response.Redirect("default.aspx")
      else
        if DataSet1.DefaultView.Table.Rows(0)("Sex")="男" then
          Sex.Items(0).Selected = True
```


图 10.26　修改代码

(3) 将代码中的 Session("userid")全部替换成 Request.Cookies("id").Value，即用 Cookie 变量 ID 来替换 Session 变量 userid。然后，保存文件即可。

“修改个人资料”页面的预览效果如图 10.27 所示。

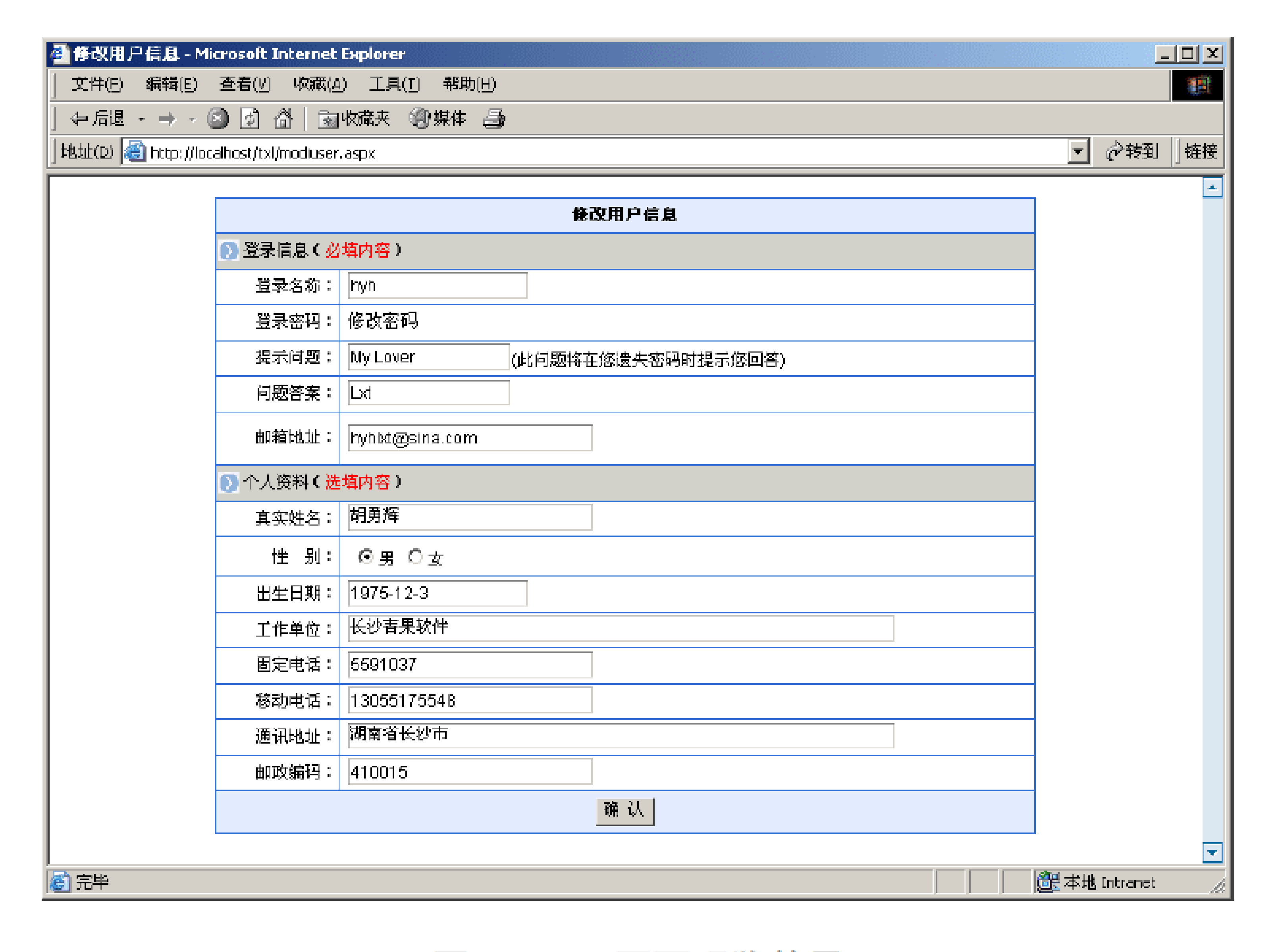


图 10.27　页面预览效果

对于本页面功能的具体实现，请参考 8.5 节，这里不再赘述。

10.3.3　我的同学录

下面介绍系统主页中的“我的同学录”功能块的实现。在“我的同学录”功能中，主要包括“我的班级”和“我的学校”两块。“我的班级”是指当前登录用户所注册的班级，而“我的学校”是指当前登录用户所注册的班级所属的学校。这里，仅使用一个表格，列出用户所注册的班级名称和所属的学校名称，并提供相应的链接。

1. 功能模块的界面设计

(1) 在表格 table2 的下面，新插入一个 3 行 3 列的表格 table4，其属性设置如图 10.28 所示。

图 10.28　表格属性

(2) 合并表格 table4 第 1 行的 3 个单元格，并在【属性】面板中设置合并单元格的背景图片为 image/bg_1.gif；合并表格 table4 第 3 行的 3 个单元格，并在【属性】面板中设置合并单元格的背景图片为 image/bg_4.gif。

(3) 分别设置第 2 行的第 1 个单元格和第 3 个单元格的背景图片为 image/bg_2.gif 和 image/bg_3.gif。

(4) 在表格 table4 的第 2 行的单元格中插入一个 9 行 5 列的表格 table5，其属性设置如图 10.29 所示。

图 10.29　表格属性

(5) 对于表格 table5 中各单元格的格式设置较为烦琐，这里不再赘述，请读者参考源文件。在表格 table5 中插入相应的文本和图片链接，其界面设计如图 10.30 所示。

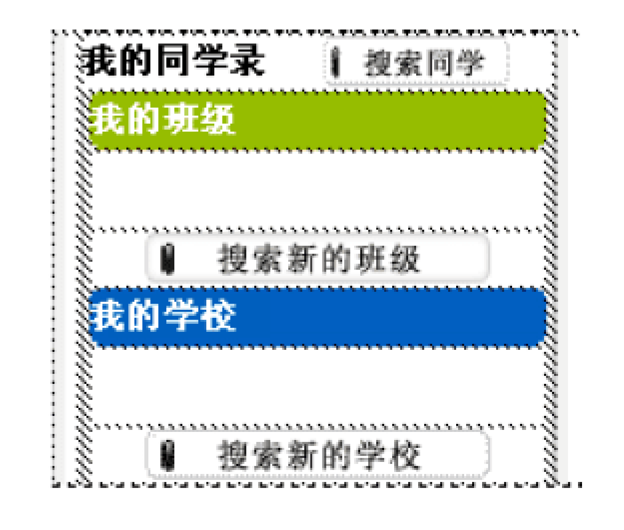
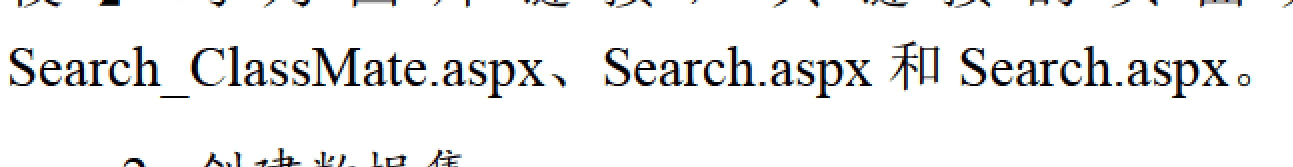


图 10.30　界面设计

其中，【搜索同学】、【搜索新的班级】、【搜索新的学校】均为图片链接，其链接的页面分别为 Search_ClassMate.aspx、Search.aspx 和 Search.aspx。



2. 创建数据集

接下来，创建两个数据集，分别用于获取“我的班级”和“我的学校”两模块所对应的动态数据。

(1) 打开【应用程序】面板组，切换至【服务器行为】面板，单击按钮，从弹出的菜单中选择【数据集】命令。在弹出的【数据集】对话框中单击【高级】按钮，切换至高

级模式。

(2) 在数据集的高级视图中，单击【添加参数】按钮[+]，在弹出的【添加参数】对话框中设置参数名称为“@UserId”，如图 10.31 所示。

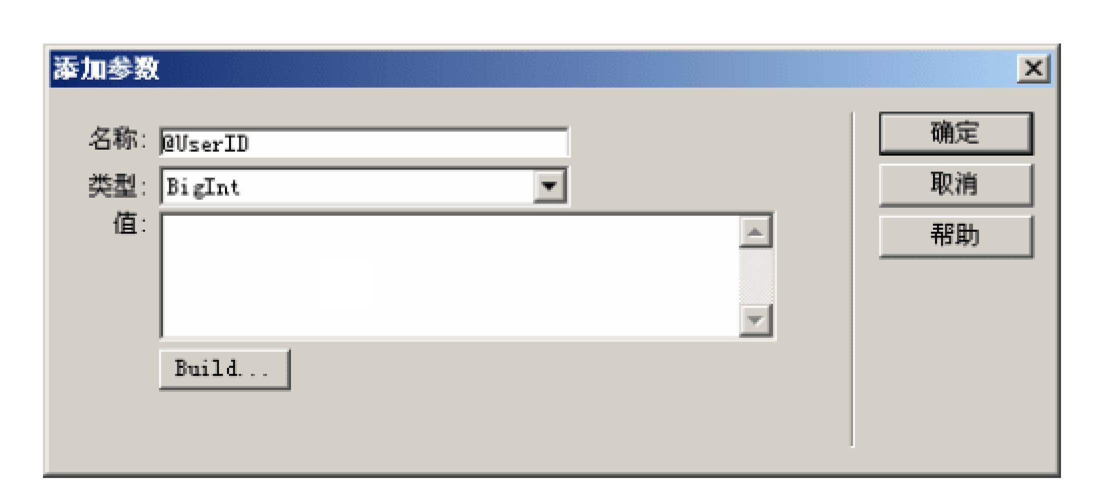


图 10.31 【添加参数】对话框

(3) 单击 Build 按钮，在弹出的【生成值】对话框中，设置参数的值为 Cookie 变量 ID，如图 10.32 所示。

图 10.32 【生成值】对话框

(4) 单击【确定】按钮，返回【添加参数】对话框；再次单击【确定】按钮，返回【数据集】对话框。设置数据集的名称为 DataSet2，连接选择 Cnn，并在 SQL 文本框中输入以下 SQL 语句：

```
select * from classinfo where id in (select classcode from userclass where
usercode=?) order by createdate
```

数据集设置界面如图 10.33 所示。

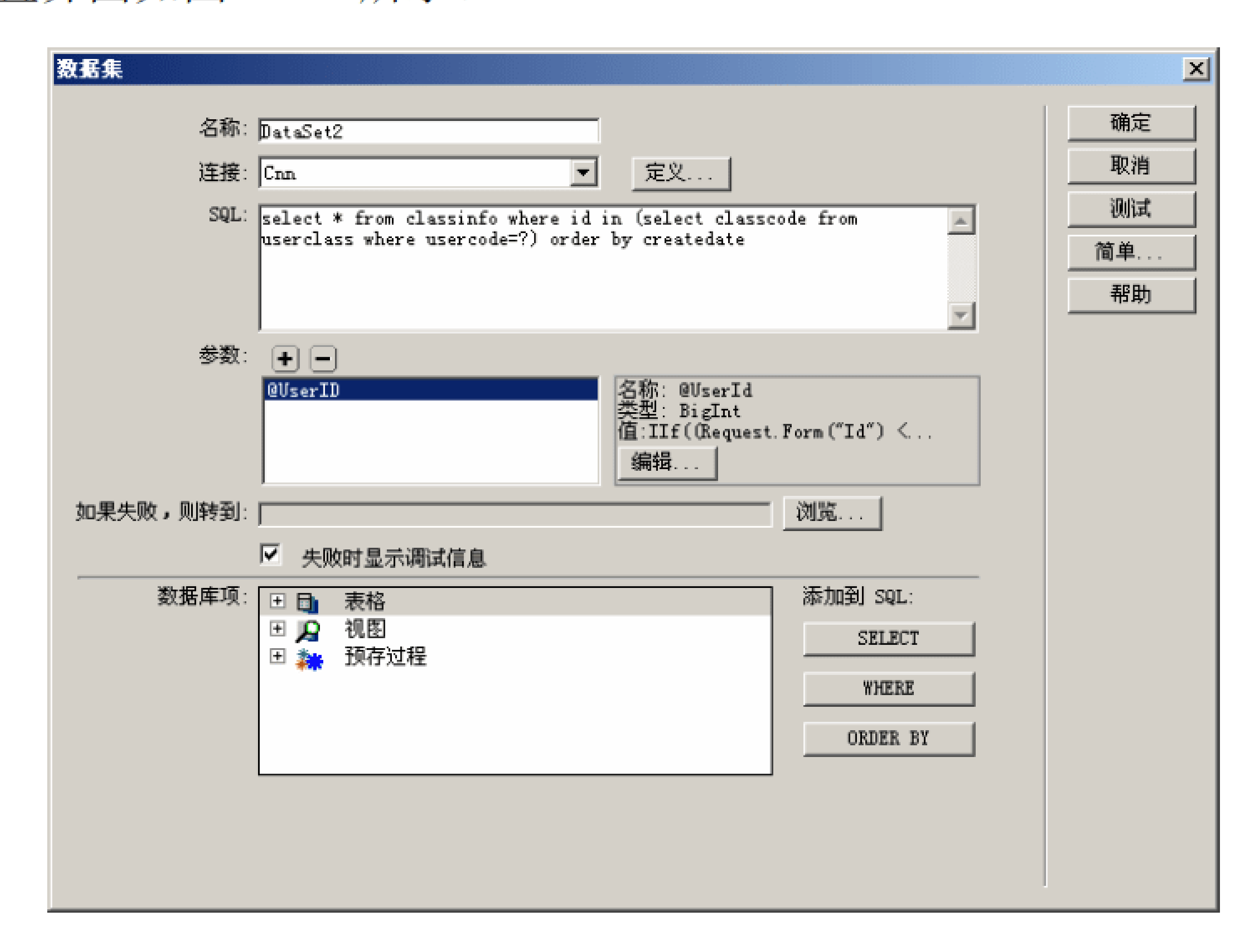


图 10.33 创建数据集 DataSet2

(5) 单击【确定】按钮，完成数据集的创建。这里所创建的数据集 DataSet2 用于获取当前登录用户所注册的班级信息。

(6) 在【服务器行为】面板中选择数据集 DataSet2 并右击，从弹出的快捷菜单中选择【拷贝】命令。然后，在【服务器行为】面板的空白处右击，从弹出的快捷菜单中选择【粘贴】命令，这样将生成一个新的数据集 DataSet3。

(7) 双击数据集 DataSet3，在弹出的【数据集】对话框中修改 SQL 语句为：

```
SELECT * FROM schoolinfo WHERE id in (select schoolid from classinfo,userclass
where classcode=classinfo.id and usercode=?) ORDER BY createdate
```

设置界面如图 10.34 所示。

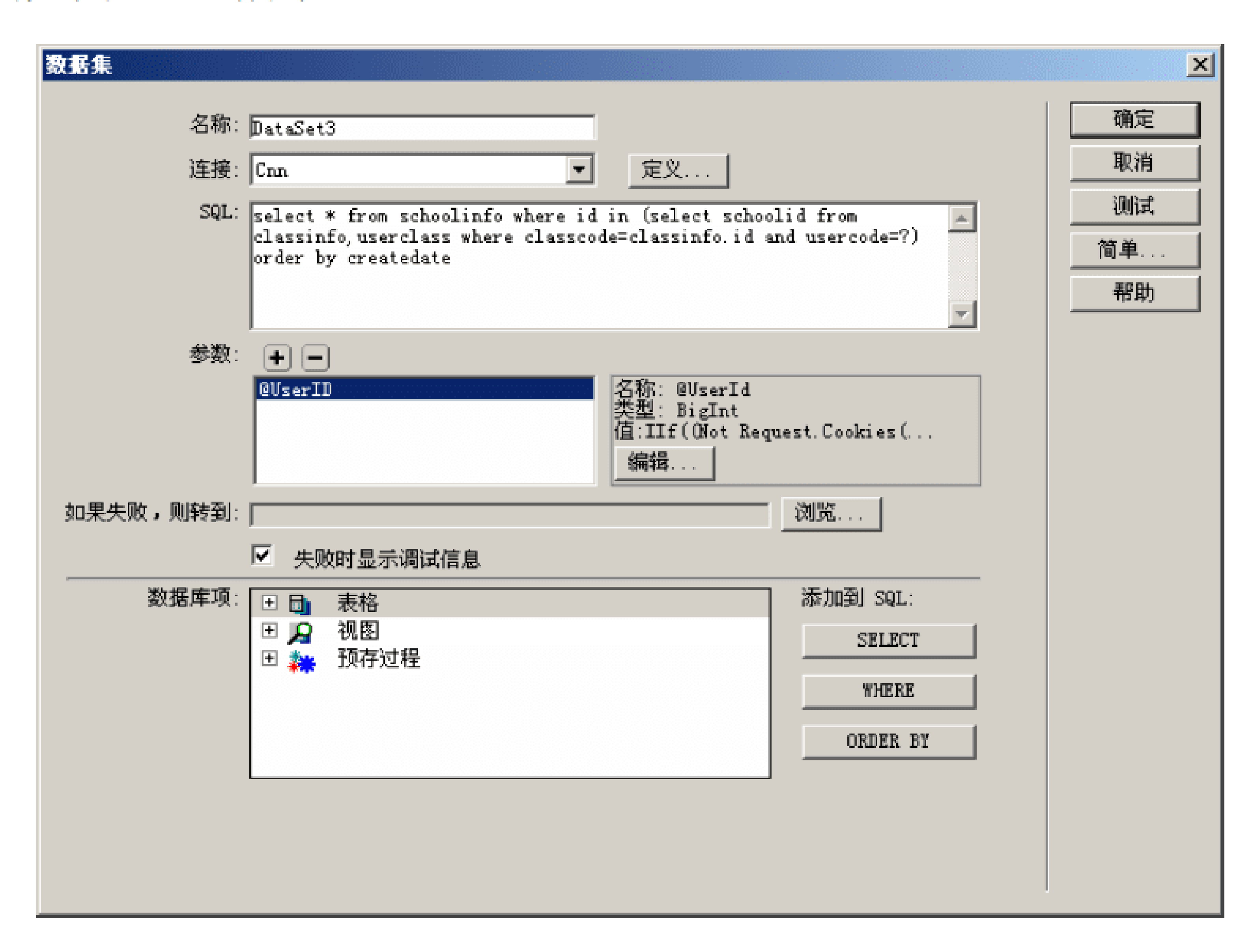


图 10.34　编辑数据集 DataSet3

(8) 单击【确定】按钮，完成数据集 DataSet3 的设置，该数据集用于获取当前登录用户所注册的班级所属的学校信息。

提示： 由于数据集 DataSet3 与数据集 DataSet2 具有相同的参数，因此可通过数据集的复制操作来快速生成新的数据集。

接下来，绑定数据。

(9) 在文本“我的班级”下的单元格中，插入一个<div>标签，并设置其相关的 style 样式，如图 10.35 所示。

```
<tr>
  <td height="30" align="left" style="padding-left:5px;">
    <div style="padding-bottom:2px;padding-left:2px;">
    </div>
  </td>
</tr>
```

图 10.35　插入<div>标签

(10) 在【应用程序】面板组中，切换至【绑定】面板，选择并展开数据集 DataSet2，

如图 10.36 所示。

(11) 选择并拖动字段 ClassName 至<div>标签中。然后，在【设计】视图中选择该动态文本，在【属性】面板中设置其【链接】属性为“main.aspx?classid=<%# DataSet2.FieldValue("id", Container) %>”，【目标】属性为_Self。

(12) 选择整个<div>标签，在【应用程序】面板组中切换至【服务器行为】面板，单击按钮，从弹出的菜单中选择【重复区域】命令。在弹出的【重复区域】对话框中，选择数据集为 DataSet2，并设置显示所有记录，如图 10.37 所示。单击【确定】按钮，完成重复区域的设置。

图 10.36 【绑定】面板

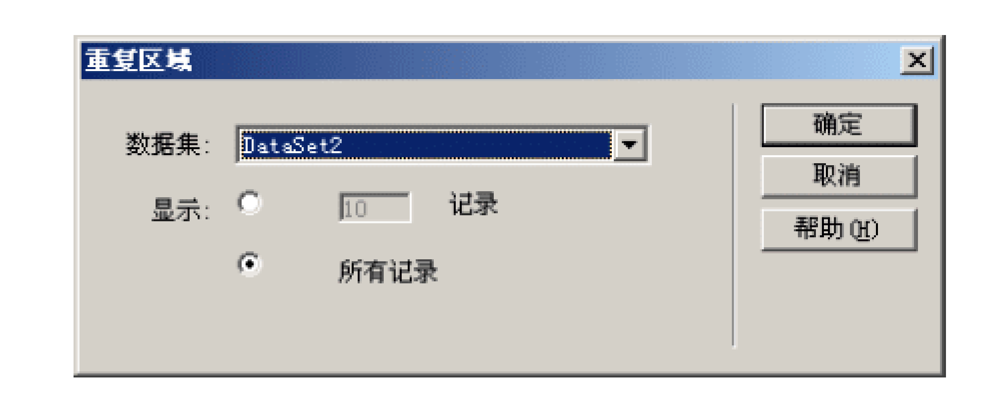


图 10.37 【重复区域】对话框

(13) 在“我的学校”下的单元格中，插入一个<div>标签，并设置其相关的 Style 样式。然后在【绑定】面板中选择并展开数据集 DataSet3，拖动字段 SchoolName 至刚才创建的<div>标签中。选择该动态文本，在【属性】面板中设置其【链接】属性为“SchoolInfo.aspx?id=<%# DataSet3.FieldValue("id", Container) %>”，【目标】属性为_Blank。

(14) 选择整个<div>标签，在【服务器行为】面板中单击按钮，在弹出的菜单中选择【重复区域】命令。在弹出的【重复区域】对话框中，选择数据集为 DataSet3，并设置显示所有记录，如图 10.38 所示。

(15) 单击【确定】按钮，完成重复区域的设置。

至此，“我的同学录”中的信息显示全部完成，其预览效果如图 10.39 所示。

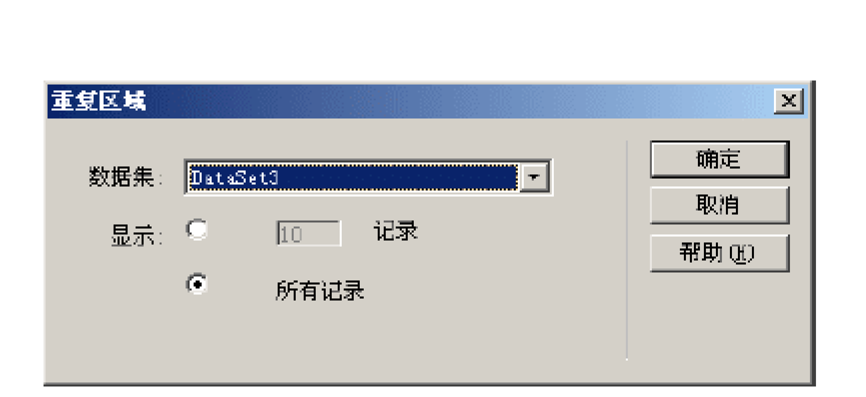


图 10.38 【重复区域】对话框

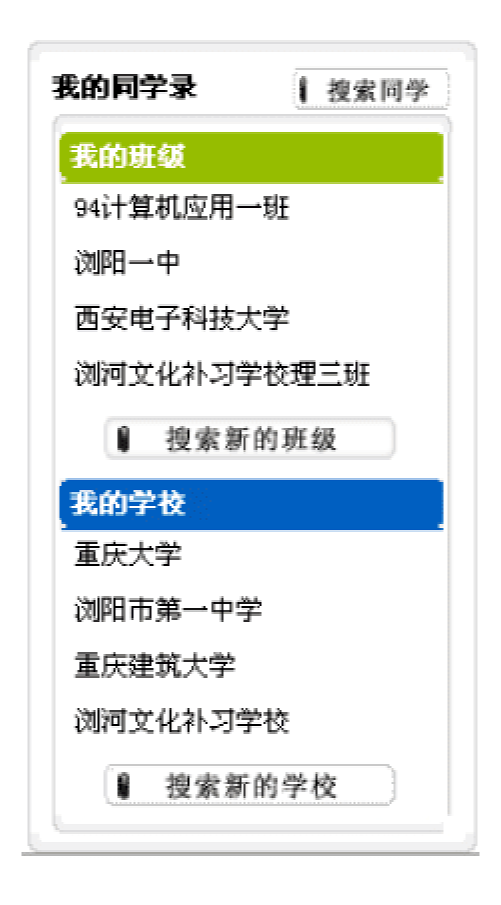


图 10.39 “我的同学录”模块的预览效果

10.3.4　班级信息

“班级信息”模块是系统主页中的一项主要功能模块，它将显示指定班级的相关信息。这里的“指定”是指通过页面参数 ClassID 所指定的班级，该参数值对应于班级信息表中的班级编码。从主页结构中可以看到，班级信息主要包括班级基本信息、最近到访的同学和班级最新相片三部分。

1. 功能模块的界面设计

首先，来看功能模块的界面设计，如图 10.40 所示。

由于班级信息的功能界面较为复杂，这里不再对每个表格的设计进行详细的介绍。图中，班级留言、班级相册、通讯录和班级管理为四个图片链接，其链接的页面分别为 Bjly.aspx、Bjxc.aspx、Bjtxl.aspx 和 Bjgl.aspx。此外，在图中还添加了一个 LinkButton 控件，该控件的显示文本为“退出此班级”，其单击事件为 ExitClass_Click。

图 10.40　界面设计

2. 创建相应的数据集

接下来创建相应的数据集，用于实现各项班级信息的动态显示。

(1) 打开【应用程序】面板组，切换至【服务器行为】面板，单击 按钮，在弹出的菜单中选择【数据集】命令。在弹出的【数据集】对话框中，单击【高级】按钮，切换至【数据集】对话框的高级模式。

(2) 在【数据集】对话框的高级模式中，单击【添加参数】按钮 ，在弹出的【添加参数】对话框中设置参数名称为“@ClassID”，如图 10.41 所示。

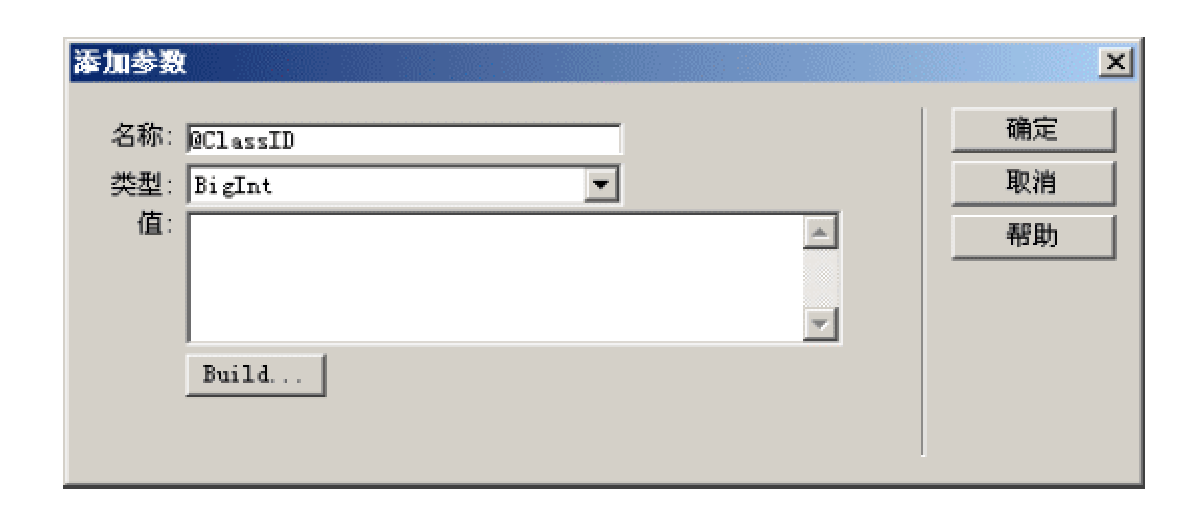


图 10.41　【添加参数】对话框

(3) 单击 Build 按钮，在弹出的【生成值】对话框中，设置参数的值为 URL 参数 ClassID，如图 10.42 所示。

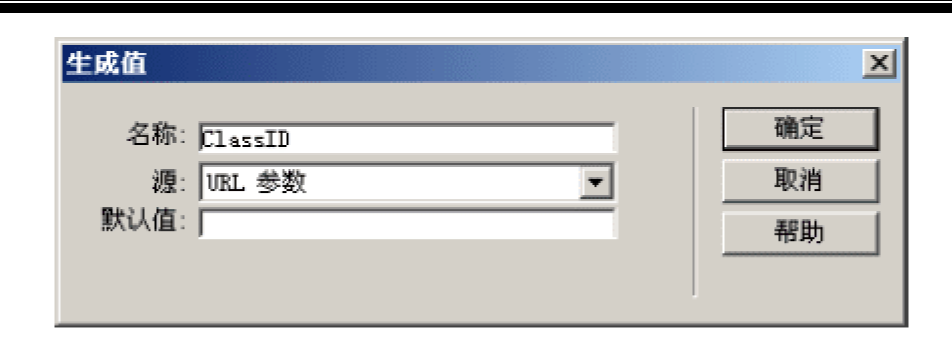


图 10.42　【生成值】对话框

(4) 单击【确定】按钮，返回【添加参数】对话框。再次单击【确定】按钮，返回【数据集】对话框。设置数据集的名称为 DataSet4，连接选择 Cnn，并在 SQL 文本框中输入以下 SQL 语句：

```
SELECT
classinfo.id,classname,grade,classinfo.createdate,schoolname,count(*)as
thenum FROM classinfo,schoolinfo,userclass WHERE schoolid=schoolinfo. id
and classcode=classinfo.id and classinfo.id=? GROUP BY
classinfo.id,classname, grade,classinfo.createdate,schoolname
```

此时的【数据集】对话框如图 10.43 所示。

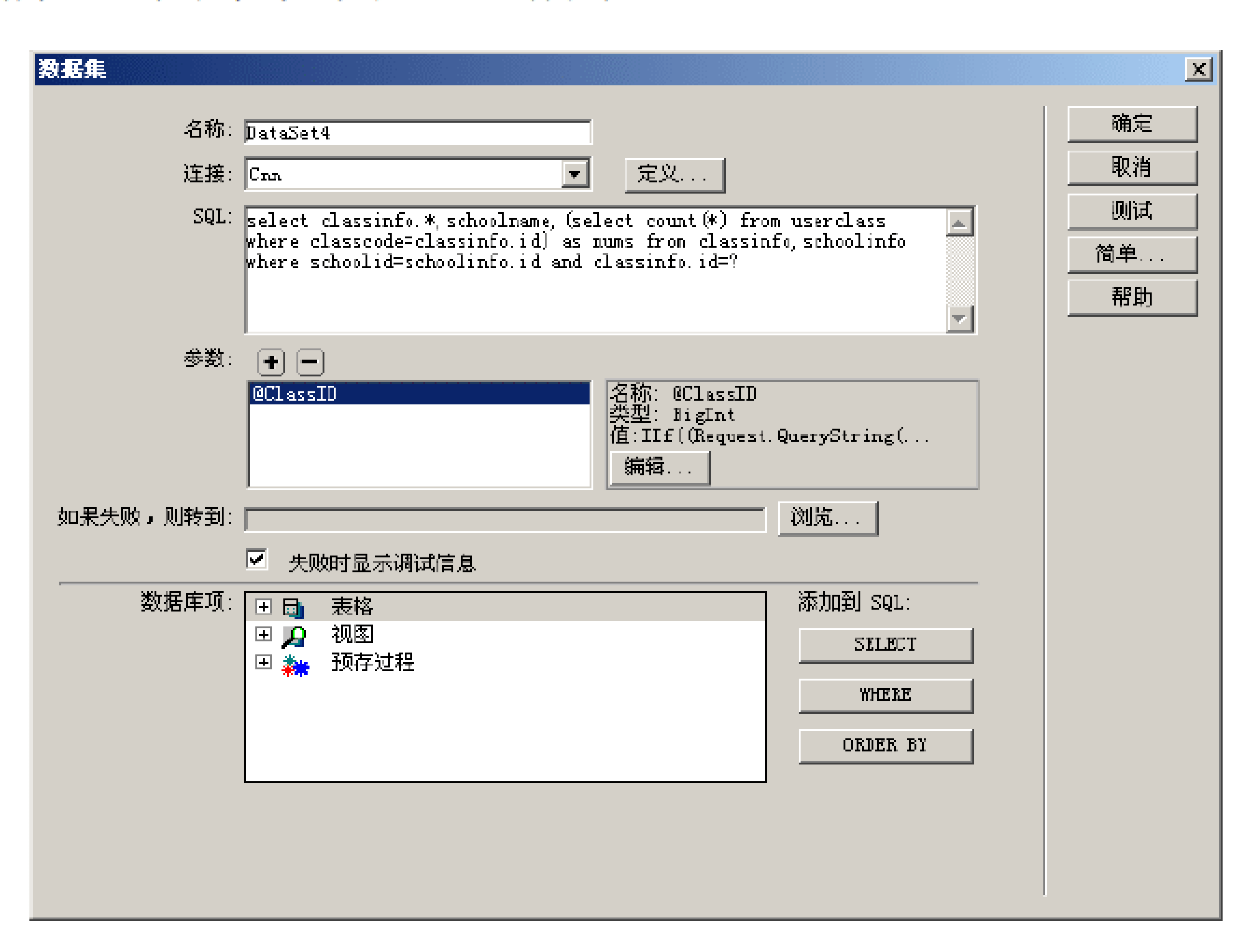


图 10.43　创建数据集 DataSet4

(5) 单击【确定】按钮，完成数据集的创建，该数据集用于获取指定班级的相关信息。

(6) 在【应用程序】面板组中切换至【绑定】面板，选择并展开数据集 DataSet4，如图 10.44 所示。

图 10.44　展开数据集 DataSet4

(7) 选择并拖动字段 SchoolName 至文本“中国同学录 >”之后；选择并拖动字段 ClassName 至文本“中国同学录 >>”之后，并在【属性】面板中设置动态文本的字体颜色为红色。重复以上操作，分别选择并拖动字段 SchoolName、CreateDate、Grade 和 TheNum 至文本“所属学校”、“创建时间”、“入学年份”以及“班级成员”之后，同时分别在【属性】面板中设置动态文本的字体颜色为红色。

(8) 选择“班级留言”图片，在【属性】面板中设置其链接为“bjly.aspx?id=<%# DataSet4.FieldValue("id", Container) %>”；选择“班级相册”图片，在【属性】面板中设置其链接为“bjxc.aspx?id=<%# DataSet4.FieldValue("id", Container) %>”；选择“通讯录”图片，设置其链接为“bjtxl.aspx?id=<%# DataSet4.FieldValue("id", Container) %>”；选择“班级管理”图片，设置其链接为“bjgl.aspx?id=<%# DataSet4.FieldValue("id", Container) %>”。

此时，数据集 DataSet4 的数据绑定已经完成。

(9) 在【应用程序】面板组中切换至【服务器行为】面板，选择数据集 DataSet4 并右击，从弹出的快捷菜单中选择【拷贝】命令。然后，在【服务器行为】面板的空白处右击，从弹出的快捷菜单中选择【粘贴】命令，这样将生成一个新的数据集 DataSet5。

(10) 双击数据集 DataSet5，在弹出的【数据集】对话框中修改 SQL 语句为：

```
SELECT username FROM usersheet,classinfo WHERE shtag='2' and
adminuser=usersheet.code and classinfo.id=?
```

此时的数据集对话框如图 10.45 所示。

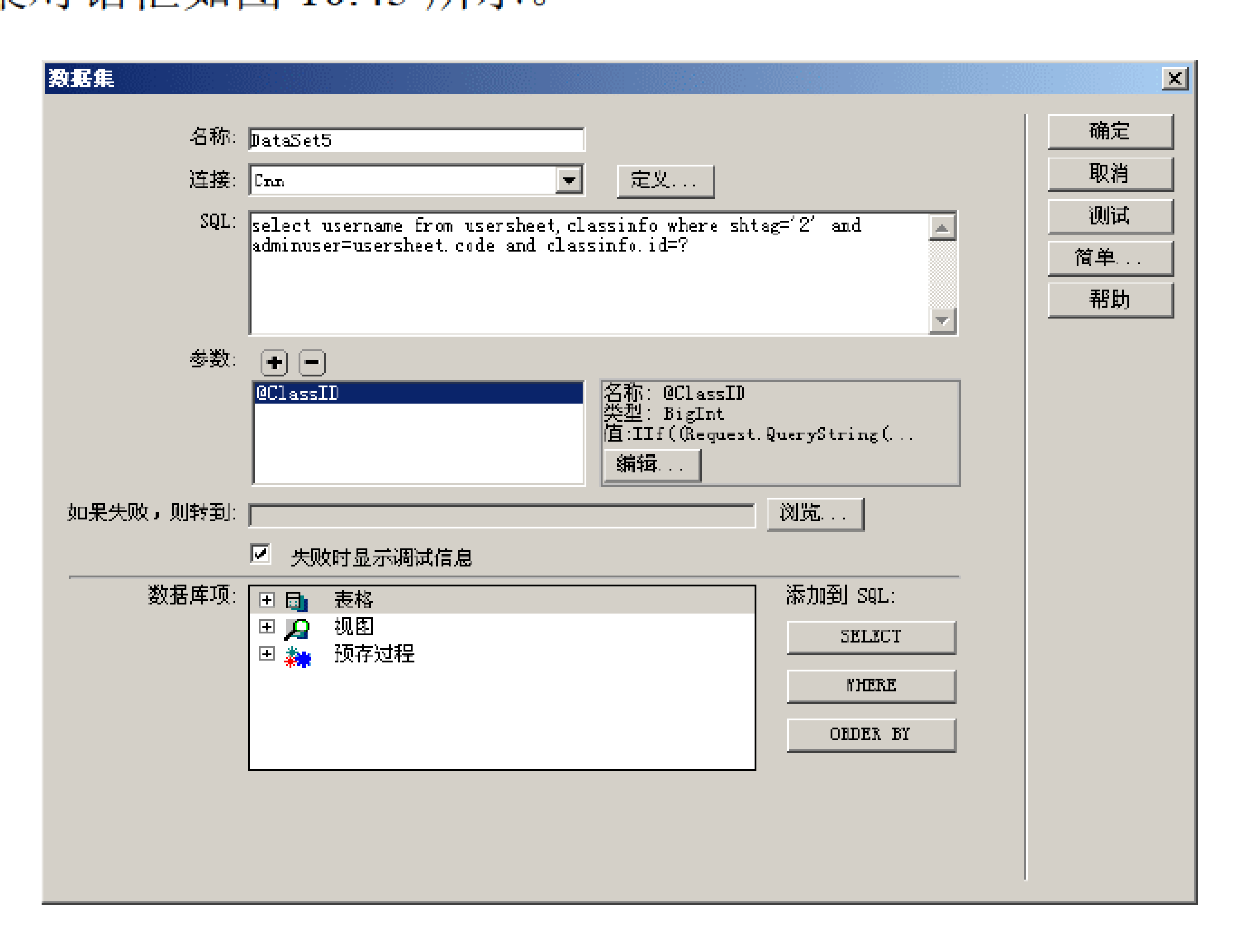


图 10.45　编辑数据集 DataSet5

(11) 单击【确定】按钮，完成数据集 DataSet5 的设置，该数据集用于获取当前班级的管理员信息。

(12) 在【应用程序】面板组中切换至【绑定】面板，选择并展开数据集 DataSet5，如图 10.46 所示。

(13) 选择并拖动字段 UserName 至文本“管理员”之后，并在【属性】面板中设置动态文本的字体颜色为红色。

(14) 复制数据集 DataSet5，并粘贴生成新的数据集 DataSet6。双击数据集 DataSet6，在弹出的【数据集】对话框中修改 SQL 语句为：

图 10.46　展开数据集 DataSet5

```
select top 10 * from usersheet where code in (select usercode from userclass
```

```
where classcode=?) order by currlogin desc
```

设置界面如图 10.47 所示。

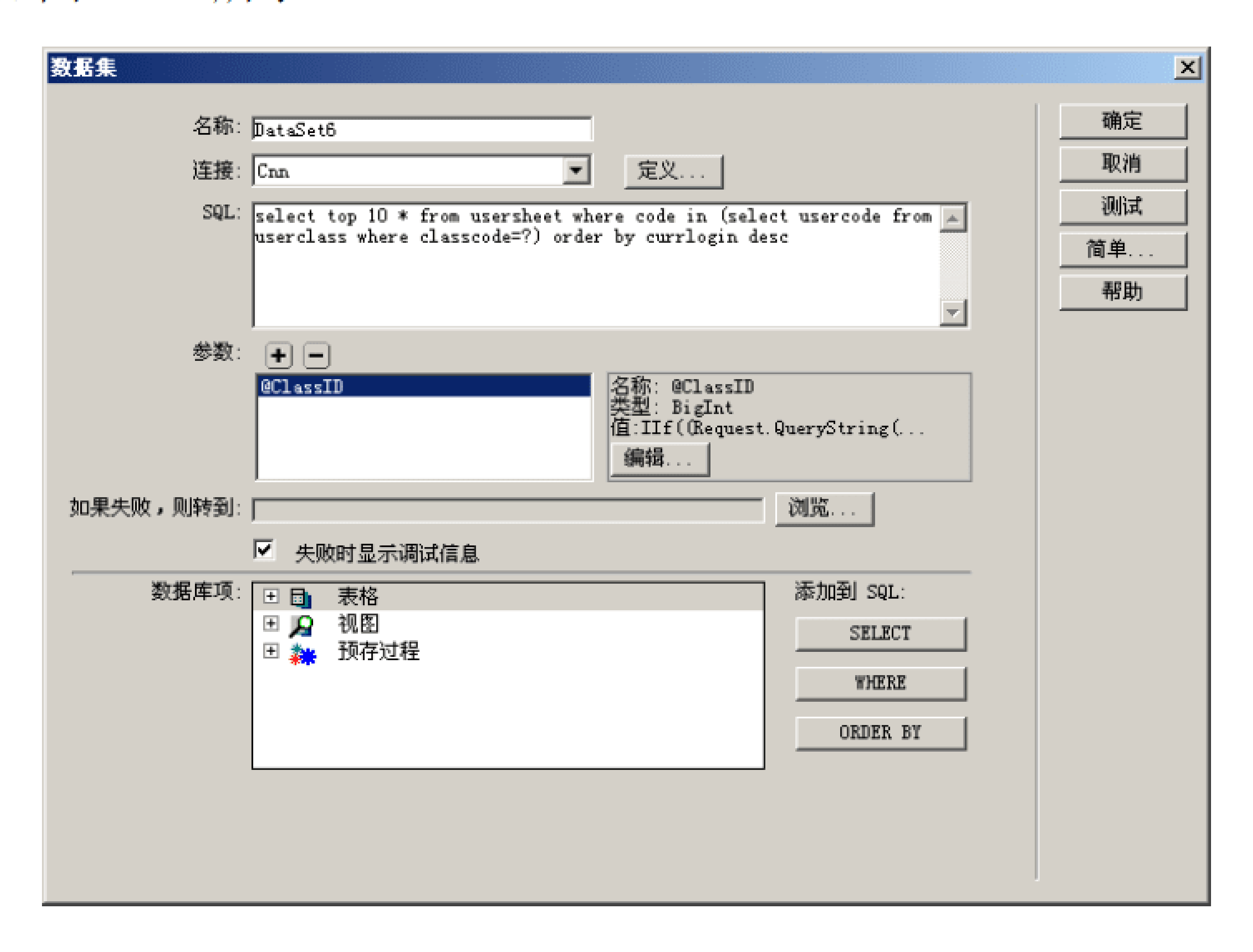


图 10.47　编辑数据集 DataSet6

(15) 单击【确定】按钮，完成数据集 DataSet6 的设置，该数据集用于获取最新到访同学的相关信息。

(16) 对于最新到访同学的信息列表，不能使用简单的动态文本，而需采用【数据列表】对话框来实现。首先，在【应用程序】面板组中将选项切换至【绑定】面板，选择并展开数据集 DataSet6，如图 10.48 所示。

图 10.48　展开数据集 DataSet6

(17) 选择并拖动字段 UserName 至文本“最近到访”下面的单元格中，并在【属性】面板中设置其字体颜色为红色，【链接】属性为“bjtxl.aspx?id=<%# DataSet4.FieldValue("id", Container) %>#<%# DataSet6.FieldValue("code", Container) %>”，【目标】属性为_blank。然后，在该动态文本之后，输入文本“(次)”，并拖动字段 Nums 至括号内。

(18) 切换至【代码】视图，选择在文本“最近到访”下面的单元格中所插入的所有代

码并剪切。在【应用程序】面板组中切换至【服务器行为】面板，单击按钮，从弹出的菜单中选择【数据列表】命令。在弹出的【数据列表】对话框中，设置数据集为 DataSet6，并设置显示所有记录。在【模板】列表框中选择【项目】，并在【内容】文本框中粘贴刚才所剪切的代码，如图 10.49 所示。

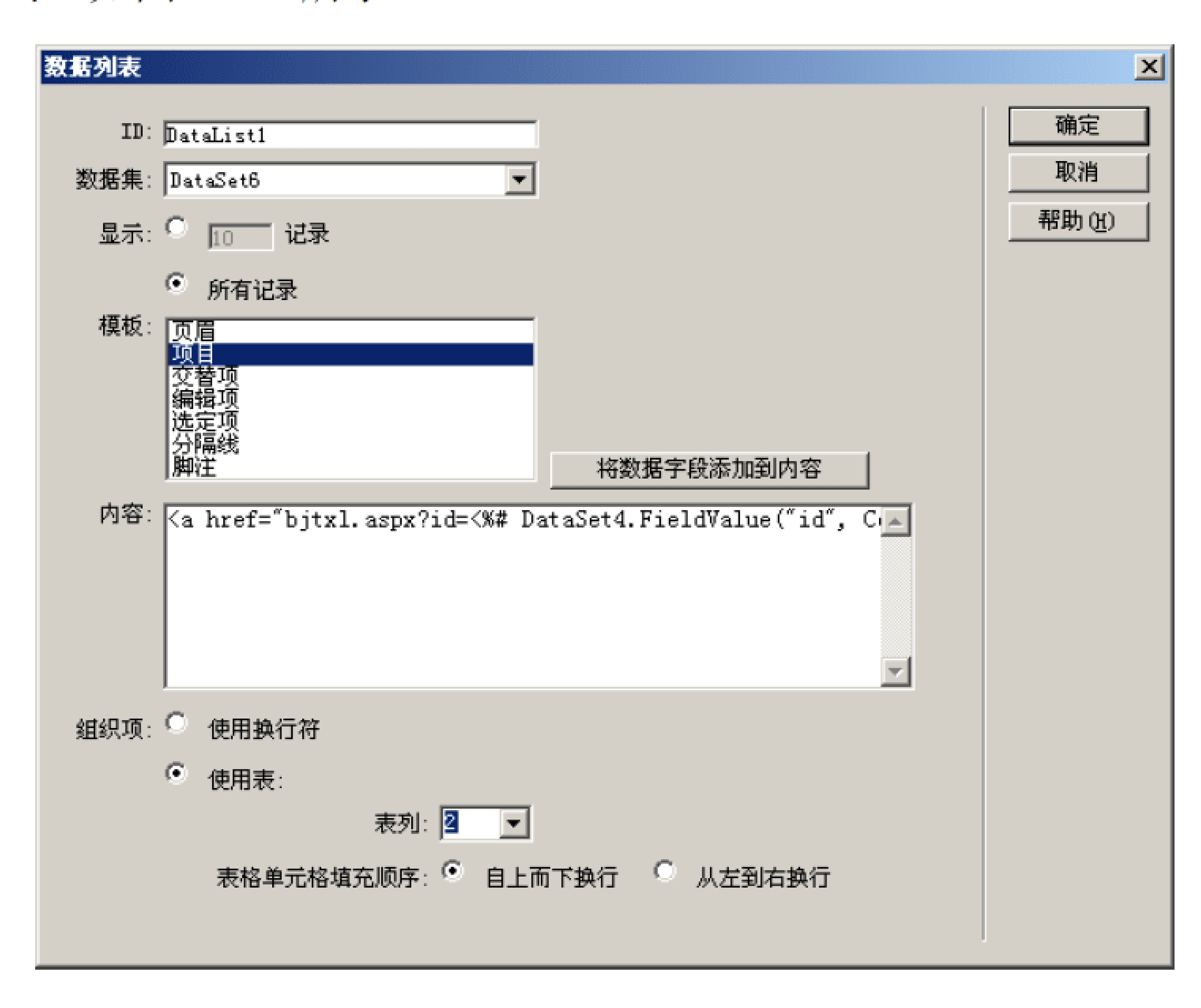


图 10.49　创建数据列表 DataList1

(19) 单击【确定】按钮，完成数据列表的创建。同时，在【设计】视图选择数据列表 DataList1 并右击，从弹出的快捷菜单中选择【编辑标签】命令。在弹出的【标签编辑器】对话框中选择【布局】分类，设置其【宽度】选项为 95%，【单元格边距】选项为 3，如图 10.50 所示。

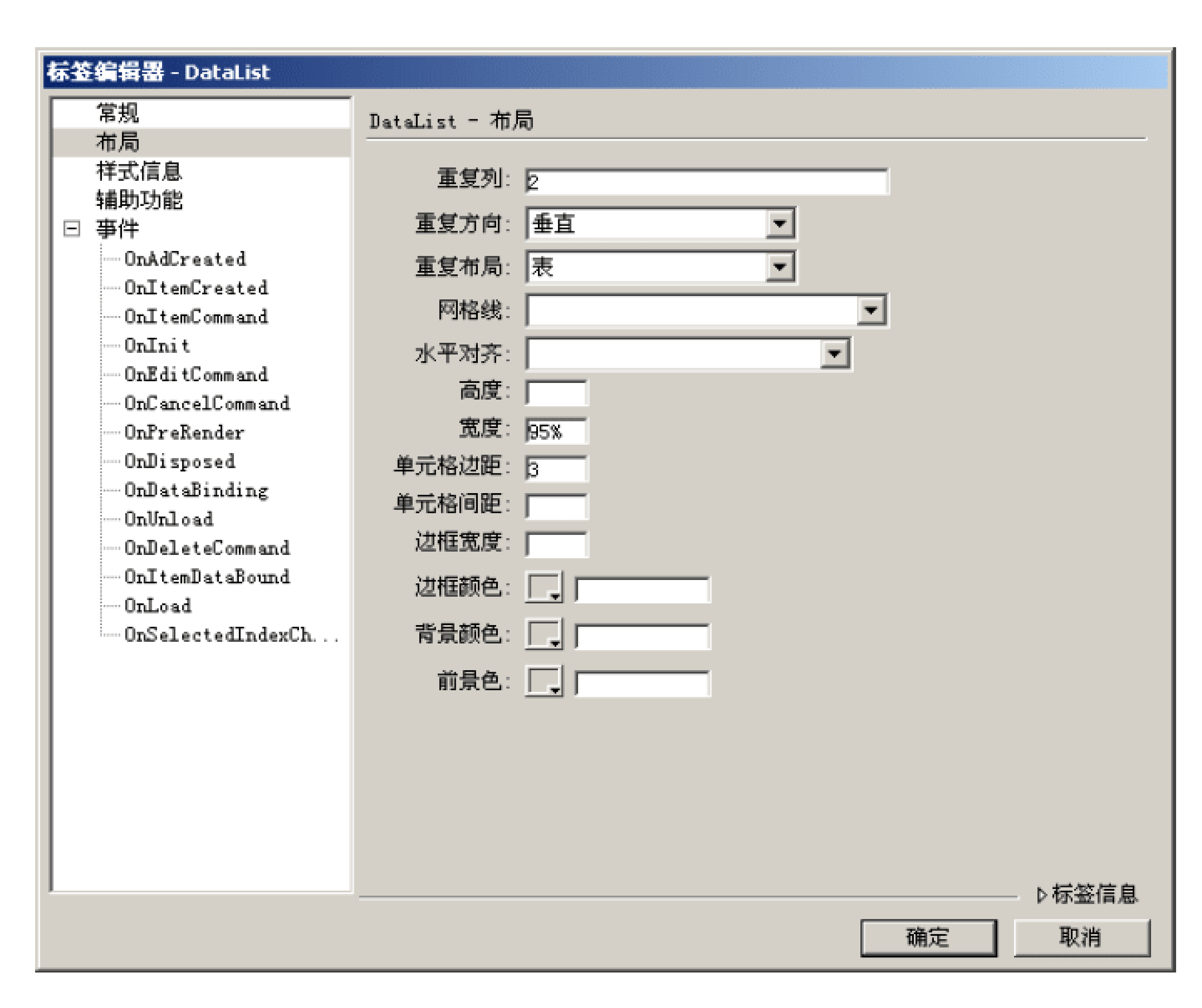


图 10.50　【标签编辑器】对话框

单击【确定】按钮，完成数据列表的设置，同时也完成了最近到访同学信息的数据绑定。

(20) 复制数据集 DataSet6，并粘贴以生成新的数据集 DataSet7。双击数据集 DataSet7，在弹出的【数据集】对话框中修改 SQL 语句为：

```
select top 4 *,('~/photo/'+photoaddr) as photo_addr from photoinfo where
classcode =? order by update desc
```

设置界面如图 10.51 所示。

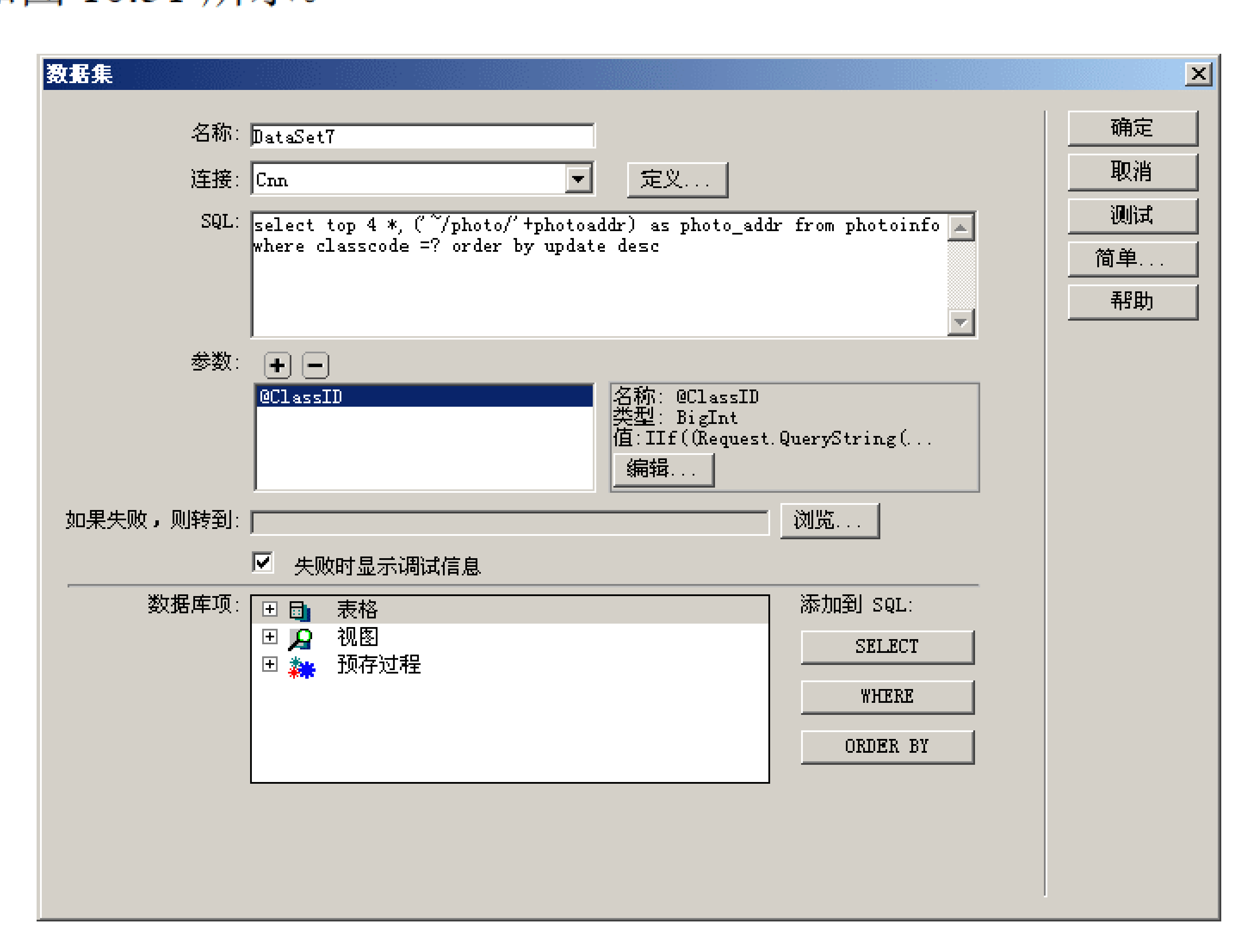


图 10.51　编辑数据集 DataSet7

(21) 单击【确定】按钮，完成数据集 DataSet7 的设置，该数据集用于获取最新班级相片的相关信息。

与最近到访同学的数据绑定类似，对于最新班级相片的数据绑定同样需要使用【数据列表】对话框来实现。

3. 创建单条数据的显示

(1) 在文本“最新相片”下的单元格中插入一个 2 行 1 列的表格，其表格属性设置如图 10.52 所示。

图 10.52　表格属性

(2) 在表格第 1 行的单元格中，插入一个 Image 控件，设置其 ImageUrl 属性为“<%# DataSet7.FieldValue ("Photo_Addr", Container) %>”，其高度为 90 像素，宽度为 130 像素。同时，在【属性】面板中设置其链接为“Viewphoto.aspx?id=<%# DataSet7.FieldValue("id", Container) %>”。

(3) 在【应用程序】面板组中切换至【绑定】面板，选择并展开数据集 DataSet7，如图 10.53 所示。

(4) 选择并拖动字段 PhotoName 至前面所创建的表格的第 2 个单元格中，然后设置两个单元格的水平对齐方式为【居中对齐】。此时，单条数据的绑定完成。

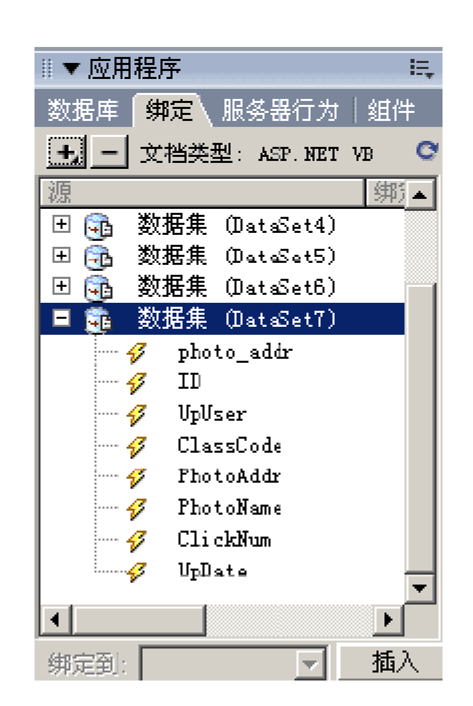


图 10.53　展开数据集 DataSet7

(5) 切换至【代码】视图，选择整个表格的代码，将其剪切。然后，在【应用程序】面板组中切换至【服务器行为】面板，单击 + 按钮，在弹出的菜单中选择【数据列表】命令。在弹出的【数据列表】对话框中，设置数据集为 DataSet7，并设置显示所有记录。在【模板】列表框中选择【项目】，并在【内容】文本框中粘贴刚才所剪切的代码，如图 10.54 所示。

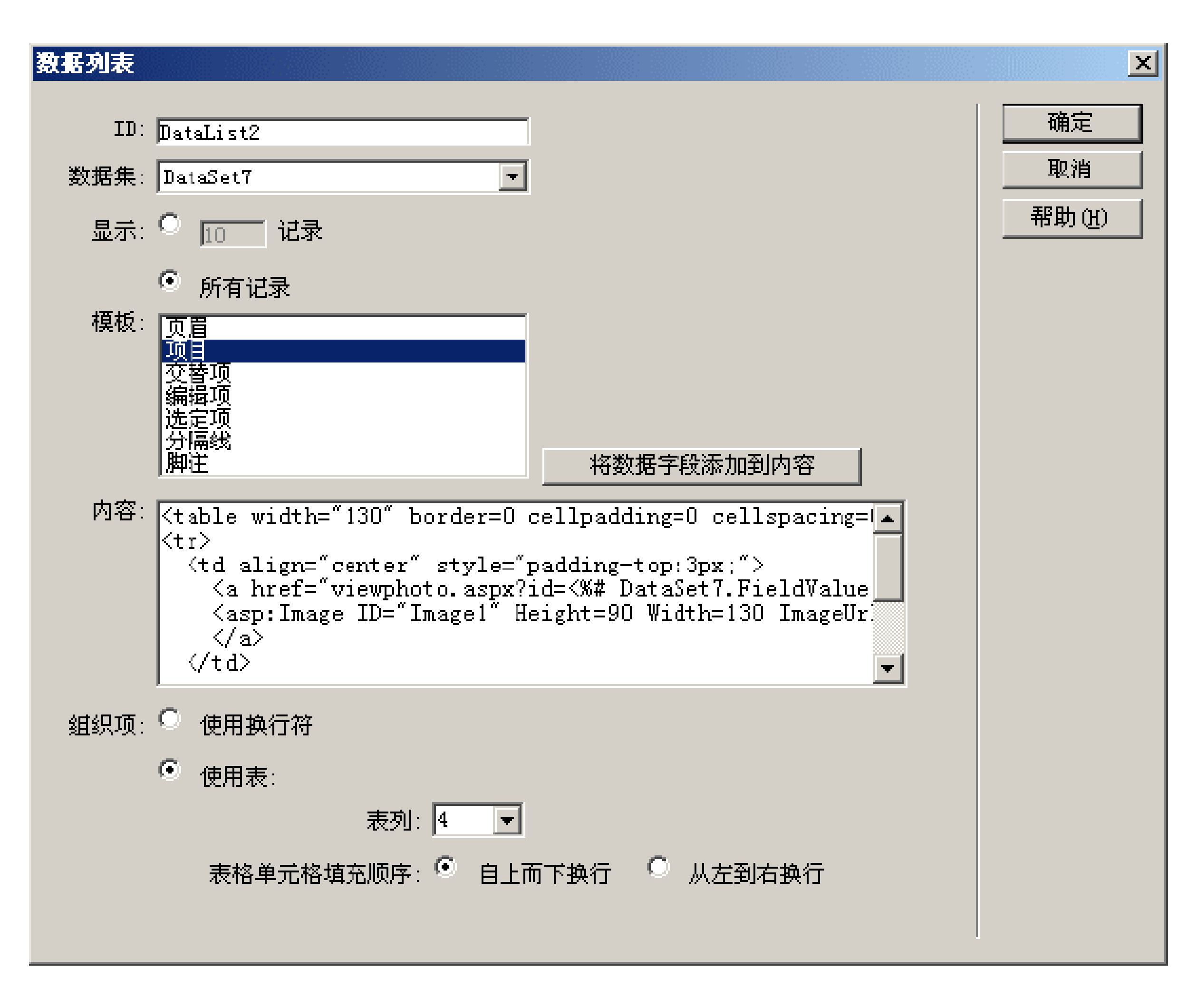


图 10.54　创建数据列表 DataList2

(6) 单击【确定】按钮，完成数据列表的创建。

至此，数据绑定工作已经完成，剩下的是相关事件代码的添加。

(7) 添加【退出此班级】链接的单击事件代码。在退出班级的操作中，为了保证数据的完整性，需要执行以下 4 步操作。

① 删除当前用户在所要退出的班级中上传的相片时发表的所有评论信息；

② 删除当前用户在所要退出的班级中上传的相片信息；

③ 删除当前用户与所要退出的班级的关联信息；

④ 如果当前用户为所要退出的班级的管理员，则还需将该班级的管理员信息清空。

以下，是【退出此班级】链接的单击事件代码：

【示例代码】

```
Sub ExitClass_Click(ByVal Sender As Object, ByVal E As EventArgs)
  Dim Cnn As OleDbConnection
  Dim Cmd As OleDbCommand
  Dim StrCnn As String
```

```
  Dim Sql As String
  '获取所要退出的班级编码
  Dim ClassID as String=request.QueryString("classid")
  '获取当前用户编码
  Dim UserID as String=Request.Cookies("id").Value
  '获取数据库连接字符串
  StrCnn = System.Configuration.ConfigurationSettings.AppSettings
("MM_CONNECTION_STRING_Cnn")
  Cnn = New OleDbConnection(StrCnn)
  Cnn.Open()  '连接数据库
  '删除对当前用户在所要退出的班级中上传的相片时发表的评论信息
  Sql = "delete from photo_pl where photoid in (select id from photoinfo where
upuser=" & trim(userid) & " and classcode=" & trim(classid) & ")"
  Cmd = New OleDbCommand(Sql, Cnn)
  Cmd.ExecuteNonquery
  '删除当前用户在所要退出的班级中上传的相片信息
  Sql = "delete from photoinfo where upuser=" & trim(userid) & " and
classcode=" & trim(classid)
  Cmd = New OleDbCommand(Sql, Cnn)
  Cmd.ExecuteNonquery
  '删除当前用户与所要退出的班级的关联
  Sql = "delete from userclass where usercode=" & trim(userid) & " and
classcode=" & trim(classid)
  Cmd = New OleDbCommand(Sql, Cnn)
  Cmd.ExecuteNonquery
  '如果当前用户为所要退出的班级的管理员，则将该班级的管理员信息清空
  Sql = "update classinfo set shtag='0' where id=" & trim(classid) & " and
adminuser=" & trim(userid)
  Cmd = New OleDbCommand(Sql, Cnn)
  Cmd.ExecuteNonquery
  Cnn.Close()  '关闭数据库连接
  '将页面跳转至本页面
  response.Redirect("main.aspx")
End Sub
```

(8) 添加页面加载时所执行的 Page_Load 事件代码。在该事件中，首先需对当前用户的身份进行验证，即根据 Cookie 变量 ID 的值来确认用户是否已登录；其次，当页面未带参数时，需要获取默认的一个班级代码并将其作为页面参数传递至本页面。

Page_Load 事件代码如下：

【示例代码】

```
Sub Page_Load(ByVal Sender As Object, ByVal E As EventArgs)
  If Not IsPostBack() Then
    '判断 Cookie 变量 ID 是否存在
    If IsDBNull(Request.Cookies("id").Value) Or
Trim(Request.Cookies("id").Value) = "" Then
      'Cookie 变量 ID 不存在，用户为非法用户，将页面跳转至登录页面
      Response.Redirect("default.aspx")
    Else
      '判断页面参数 ClassID 是否存在
      if Request.QueryString("ClassID")="" then
```

```
        '页面参数 ClassID 为空，获取数据集 DataSet2 的第一条记录的字段 ID 的值作为默认
的班级编码
        if DataSet2.DefaultView.Table.Rows.Count>0 Then
          Response.Redirect("main.aspx?ClassID=" &
DataSet2.DefaultView.Table.Rows(0)("ID"))
        End if
      End if
    End If
  End If
End Sub
```

至此，系统主页中的所有功能全部完成，页面预览效果如图 10.55 所示。

图 10.55　页面预览效果

10.4　班 级 留 言

班级留言页面为同班同学提供了一个相互交流的场所，它与第 7 章中的留言簿较为相似。不同的是，普通留言簿是针对站点或管理员发表的留言，管理员可以对留言信息进行屏蔽或删除；而班级留言簿是班级中的所有同学相互留言，大家处于一个同等的地位，所发表的留言一般来说也无须进行管理(当然，也可由班级管理员进行管理)。

在班级留言中，主要包括留言信息的浏览和发表新留言两项功能。

10.4.1　浏览留言

(1) 新建一个 ASP.NET VB 类型的页面，将其命名为 bjly.aspx。选择【修改】|【模板】|【套用模板到页】命令，在弹出的【选择模板】对话框中选择【页面模板】。单击【确定】按钮，将模板应用到本页面。

(2) 在可编辑区域 EditRegion1 中的表单 Form 中，插入一个 1 行 2 列的表格 table2，表格属性的设置如图 10.56 所示。

图 10.56　表格属性

(3) 设置第 2 个单元格的水平对齐方式为【右对齐】，并在两个单元格中分别插入图片和文本，如图 10.57 所示。

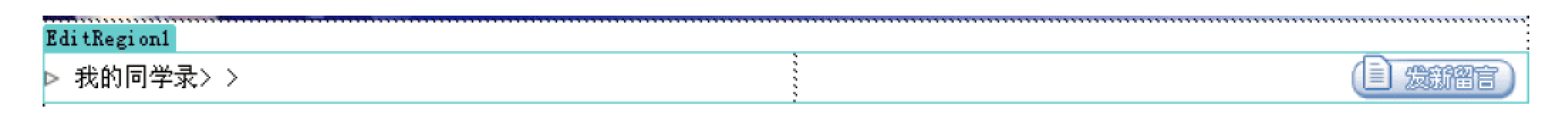


图 10.57　插入的图片和文本

(4) 选择文本“我的同学录”，在【属性】面板中设置其链接为 Main.aspx，目标为_self；选择图片“发新留言”，在【属性】面板中设置其链接为“Bjly_Add.aspx?ClassID=<%=Request.QueryString("id")%>”，目标为_self。在文本“我的同学录”后面的两个尖括号之后，分别绑定用于显示当前班级的所属学校名称以及班级名称的字段。

(5) 打开【服务器行为】面板，单击 ± 按钮，在弹出的菜单中选择【数据集】命令。在弹出的【数据集】对话框中，单击【高级】按钮，切换至【数据集】对话框的高级模式。

(6) 在【数据集】对话框的高级模式中，单击【添加参数】按钮 + ，在弹出的【添加参数】对话框中设置参数名称为“@ClassID”，如图 10.58 所示。

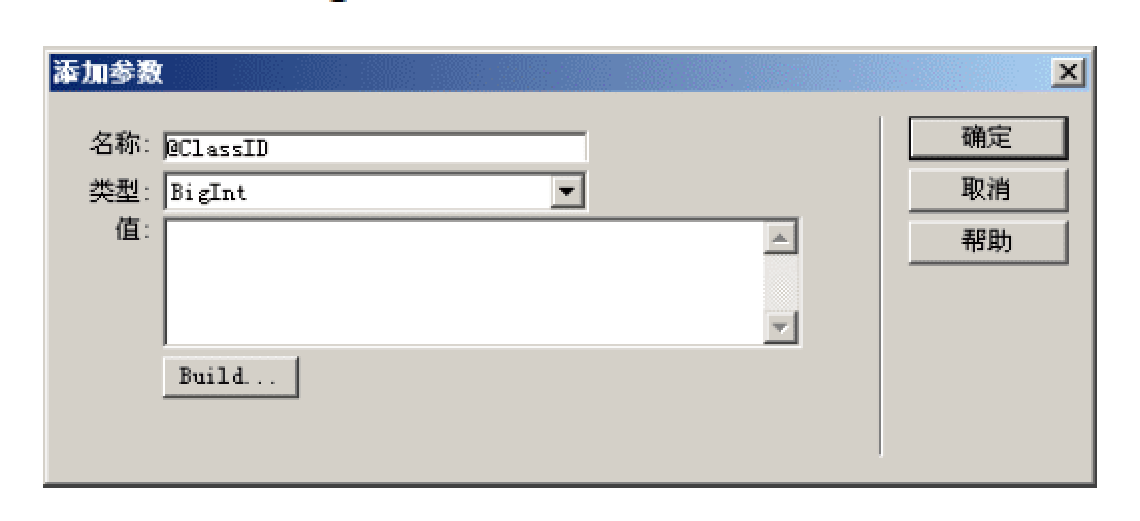


图 10.58　【添加参数】对话框

(7) 单击 Build 按钮，在弹出的【生成值】对话框中，设置参数的值为 URL 参数 ID，如图 10.59 所示。

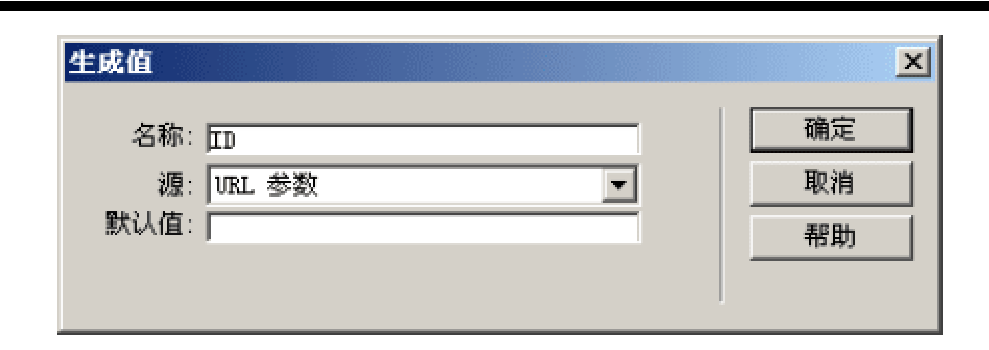


图 10.59　【生成值】对话框

(8) 单击【确定】按钮，返回【添加参数】对话框；再次单击【确定】按钮，返回【数据集】对话框。设置数据集的名称为 DataSet1，连接选择 Cnn，并在 SQL 文本框中输入以下 SQL 语句：

```
select classinfo.id,classname,schoolname from classinfo,schoolinfo where
schoolid=schoolinfo.id and classinfo.id=?
```

如图 10.60 所示。

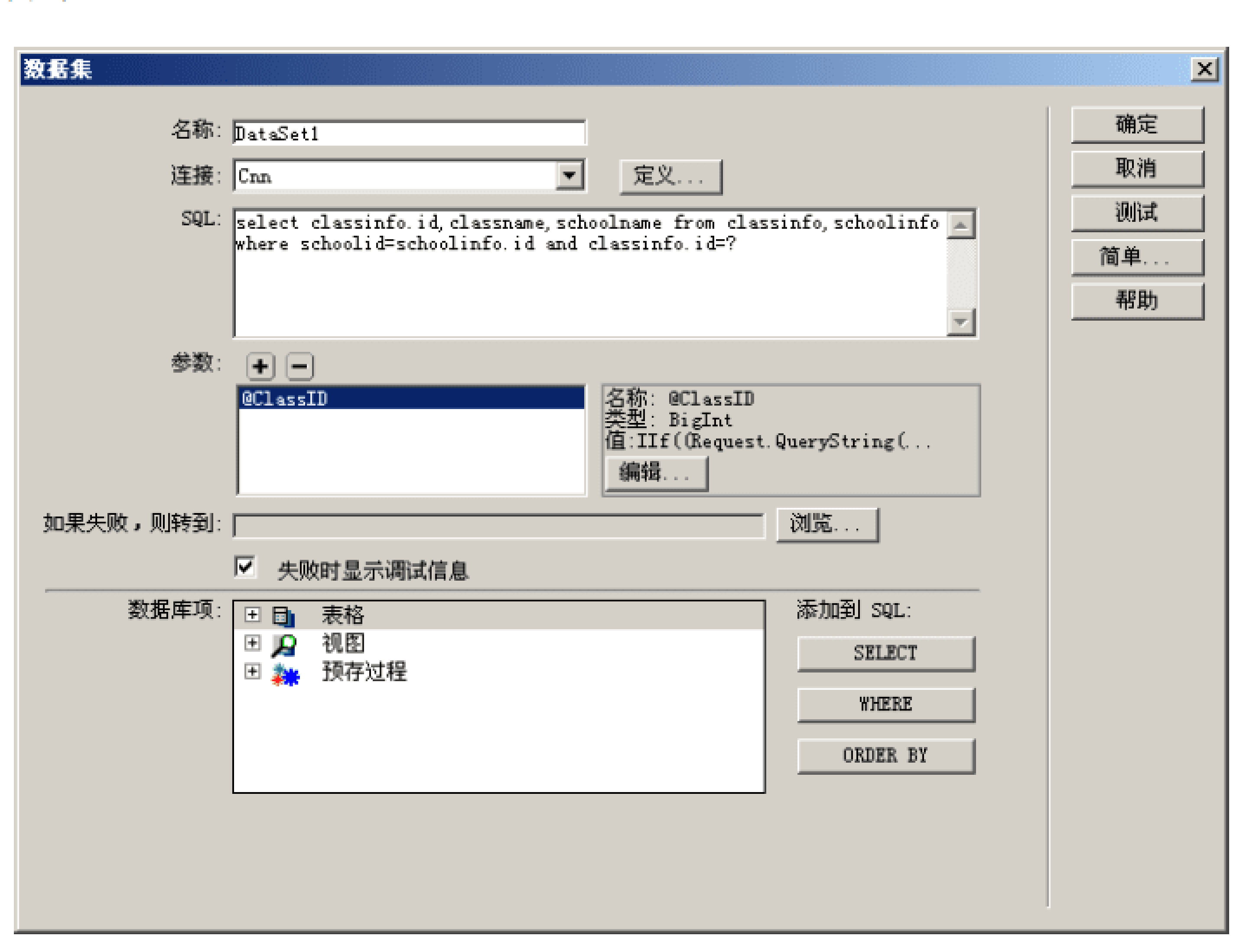


图 10.60　创建数据集 DataSet1

(9) 单击【确定】按钮，完成数据集的创建，该数据集用于获取当前的班级名称以及所属的学校名称。

(10) 在【绑定】面板中选择并展开数据集 DataSet1，如图 10.61 所示。

图 10.61　【绑定】面板

(11) 选择并拖动字段 SchoolName 至文本“我的同学录>”之后，并在【属性】面板中设置其字体颜色为红色；选择并拖动字段 ClassName 至文本“我的同学录>>”之后，并在【属性】面板中设置其字体颜色为红色。

接下来，介绍班级留言信息的浏览功能的实现，这里需要使用【服务器行为】面板中的【重复区域】命令。

(12) 创建一个数据集，以便绑定留言信息的数据显示。复制数据集 DataSet1，并粘贴生成新的数据集 DataSet2。双击数据集 DataSet2，在弹出的【数据集】对话框中修改 SQL 语句为：

```
select theno,userid,classid,title,content,ttime,username from
lyinfo,usersheet where classid=? and userid=usersheet.code order by ttime
desc
```

如图 10.62 所示。

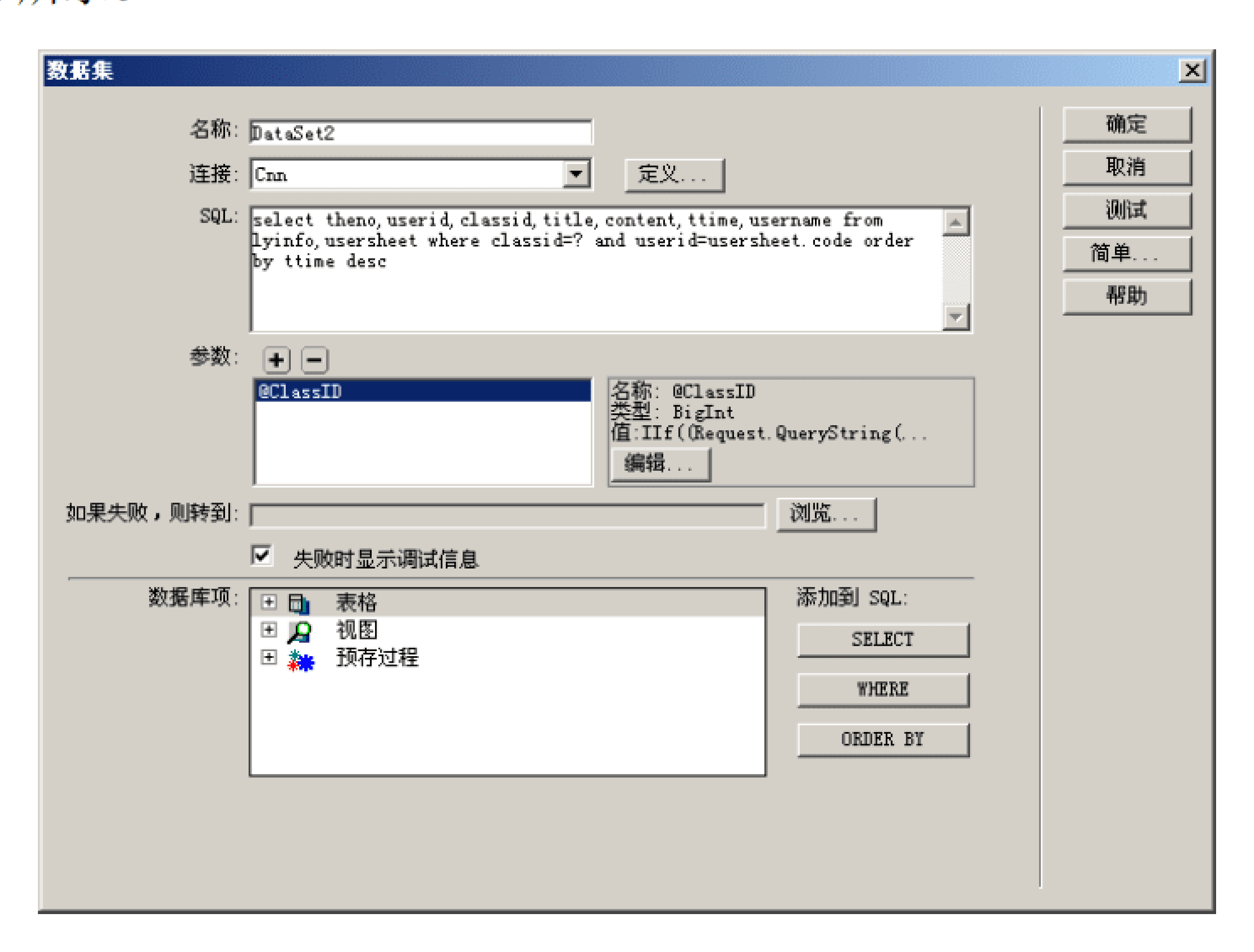


图 10.62　编辑数据集 DataSet2

单击【确定】按钮，完成数据集 DataSet2 的设置，该数据集将返回指定班级的所有留言信息。

(13) 创建单条留言数据的显示。在表格 table2 后，插入一个 4 行 4 列的表格，其表格属性设置如图 10.63 所示。设置各单元格的属性，并添加相应文本，其布局如图 10.64 所示。

图 10.63　表格属性

图 10.64　设计的表格

(14) 在【绑定】面板中，选择并展开数据集 DataSet2，如图 10.65 所示。

(15) 选择并拖动字段 Title 至文本“主题：”之后，选择并拖动字段 UserName 至文本“留言人：”的下面，选择并拖动字段 Content 至中间的单元格中，选择并拖动字段 Ttime 至文本“发表时间：”之后。然后，选择文本“查看联系方式”，在【属性】面板中设置其链接为“bjtxl.aspx?id=<%# DataSet2.FieldValue("classid", Container) %>#<%#

DataSet2.FieldValue("userid", Container) %>”，【目标】属性为_blank。接着，在【服务器行为】面板中选择并双击动态文本 DataSet2.Content，在弹出的【动态文本】对话框中，设置其格式为【编码-HTML 编码格式】，如图 10.66 所示。

图 10.65　【绑定】面板

图 10.66　设置动态文本的格式

此时，单条留言数据的绑定显示完成。

(16) 选择显示单条留言数据的整个表格，在【服务器行为】面板中单击 按钮，从弹出的菜单中选择【重复区域】命令。在弹出的【重复区域】对话框中选择 DataSet2 作为数据集，并设置显示所有记录，如图 10.67 所示。

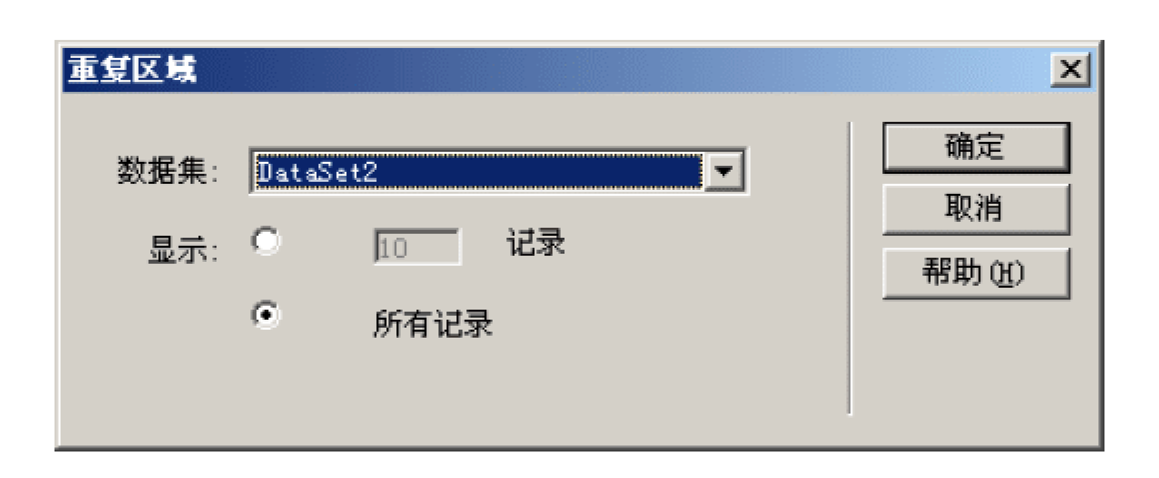


图 10.67　【重复区域】对话框

单击【确定】按钮，完成重复区域的添加。

(17) 切换至【代码】视图，添加页面的 Page_Load 事件代码，如下：

【示例代码】

```
Sub Page_Load(ByVal Sender As Object, ByVal E As EventArgs)
  If Not IsPostBack() Then
    '判断 Cookie 变量 ID 是否为空
    If IsDBNull(Request.Cookies("id").Value) Or
Trim(Request.Cookies("id").Value) = "" Then
      '将页面跳转至登录页面
      Response.Redirect("default.aspx")
    End If
  End If
End Sub
```

该事件主要用于对当前用户的身份进行验证，判断其是否已进行登录。如果用户尚未登录，则将页面跳转至登录页面。

至此，浏览留言的功能全部设计完成，其页面预览效果如图 10.68 所示。

图 10.68　页面预览效果

10.4.2　发表留言

在浏览留言页面中，单击【发新留言】图片链接，即可进入发表留言页面。在发表留言页面中，只需录入留言主题和留言内容即可。对于留言人，可根据 Cookie 变量 ID 来获取；而对于留言的班级，则可根据页面变量 ClassID 来获取。

1. 设计页面的基本结构

新建一个 ASP.NET VB 动态页，将其命名为 bjly_add.aspx。选择【修改】|【模板】|【套用模板到页】命令，在弹出的【选择模板】对话框中，选择【页面模板】。单击【确定】按钮，将模板应用到本页面。

在可编辑区域 EditRegion1 中的表单 Form 中，插入一个 3 行 1 列的表格 table2，表格属性设置如图 10.69 所示。

图 10.69　表格属性

其中，第 1 行用于显示当前班级所属的学校名称及班级名称，第 2 行用于控制第 1 行与第 3 行的间距，第 3 行则用于显示留言信息输入的表单。

这里不再对表格的设计过程进行具体的介绍，最终的布局如图 10.70 所示。

在图 10.70 所示的布局中，【提交】按钮下面的两个文本为两个 RequiredFieldValidator 验证控件，分别用于对留言主题和留言内容的输入进行验证，要求必须在两个文本框中输入内容。

图 10.70　页面设计

2. 创建一个数据集

首先需创建一个数据集，主要用于绑定学校名称与班级名称的显示。该数据集与浏览留言页面中的数据集 DataSet1 非常相似，仅参数的取值不同。可惜 Dreamweaver 8 不支持跨页面的数据集复制，因此还得重新添加。

打开【服务器行为】面板，单击+按钮，从弹出的菜单中选择【数据集】命令。在弹出的【数据集】对话框中，单击【高级】按钮，切换至【数据集】对话框的高级模式。

在【数据集】对话框的高级模式中，单击+按钮，添加一个新的参数，其名称设置为“@ClassID”，参数值设置为 URL 参数 ClassID。在【数据集】对话框中，设置数据集的名称为 DataSet1，连接选择 Cnn，并在 SQL 文本框中输入 SQL 语句：

```
select classinfo.id,classname,schoolname from classinfo,schoolinfo where
schoolid=schoolinfo.id and classinfo.id=?
```

数据集设置界面如图 10.71 所示。

图 10.71　创建数据集 DataSet1

单击【确定】按钮，完成数据集的创建，该数据集将返回当前的班级名称以及班级所属的学校名称。

在【绑定】面板展开数据集 DataSet1。选择并拖动字段 SchoolName 至文本“我的同学录>”之后，在【属性】面板中设置其字体颜色为红色；选择并拖动字段 ClassName 至文本“我的同学录>>”之后，在【属性】面板中设置其字体颜色为红色。然后，设置文本“我的同学录”链接至 Main.aspx 文件。

3. 留言信息的提交功能

(1) 实现留言信息的提交功能。打开【应用程序】|【服务器行为】面板，单击+按钮，在弹出的菜单中选择【插入记录】命令。在【插入记录】对话框中，连接选择 Cnn，所要插入的表格选择 LyInfo，提交成功后所转向的页面设置为“bjly.aspx?id=<%# Request.querystring("classid")%>”。

(2) 设置各列所对应的获取值。这里需要设置的列包括 Title、Content、ClassID 和 UserID。其中，列 Title 和 Content 的获取值分别设置为表单变量 title 和 content 的值。此时需要考虑的是，如何设置列 ClassID 和 UserID 的获取值，因为它们并不是从表单变量中获取，而是分别获取页面参数 ClassID 和 Cookie 变量 ID 的值。但在【插入记录】对话框中，只能设置从表单变量中获取值。那么这个问题怎么解决呢？一般来说，有两种方法。一种是在页面中添加隐藏域，来存储在插入记录中需要获取的值，另一种方法是修改 Dreamweaver 8 中【插入记录】对话框对应的代码。在前面的章节中，采用了第一种方法。这里，将采用第二种方法。为了便于修改代码，将列 ClassID 和 UserID 的获取值均设置为表单变量 Title，如图 10.72 所示。

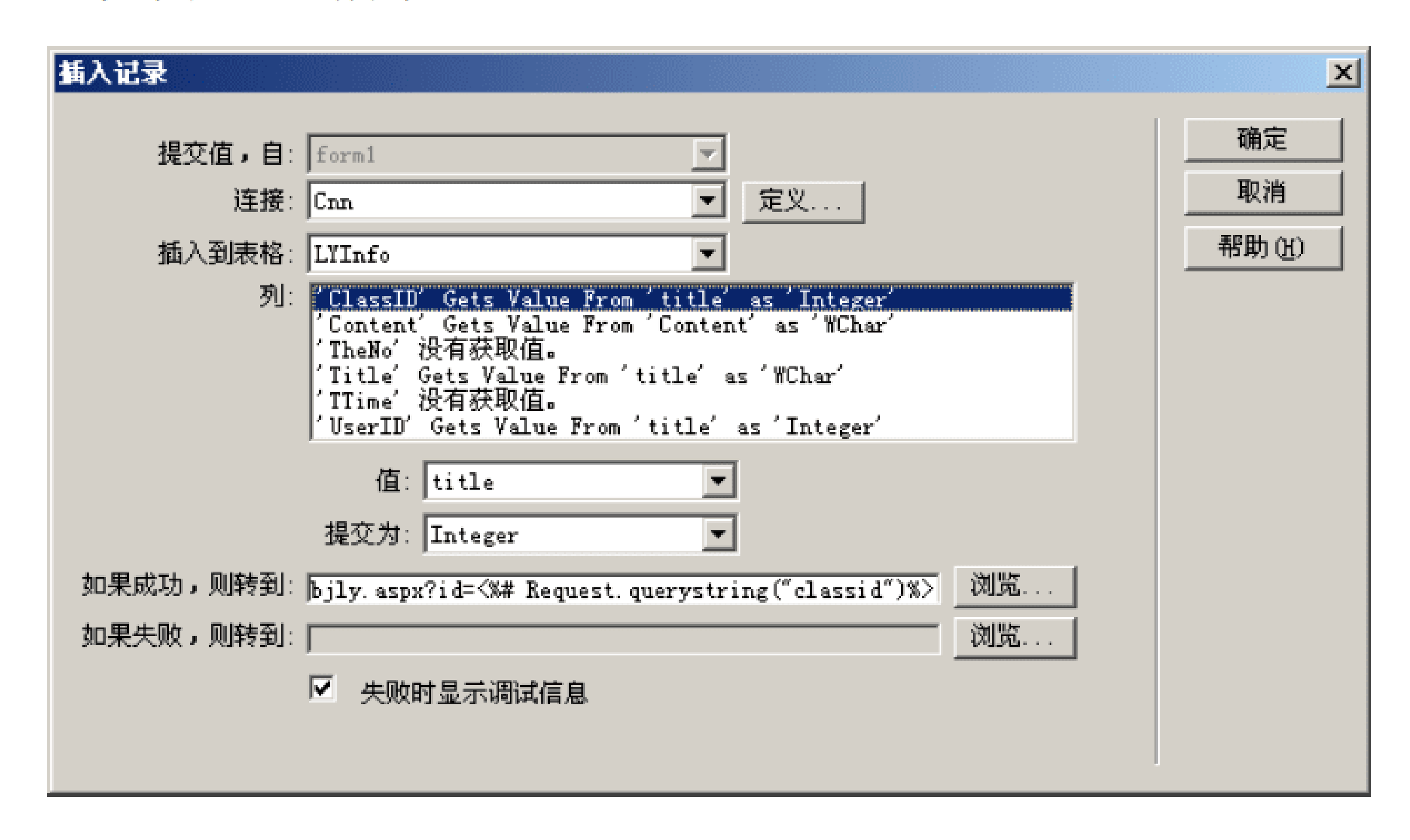


图 10.72　【插入记录】对话框

(3) 单击【确定】按钮，完成插入记录的添加。

(4) 切换至【代码】视图，找到【插入记录】对话框对应的代码，如图 10.73 所示。

修改变量“@ClassID”的取值，将表达式“Request.Form("title")”替换成“Request.querystring("Classid")”；然后修改变量“@UserID”的取值，将表达式“Request.Form("title")”替换成“Request.Cookies("id").value”。

```
<MM:Insert
runat="server"
CommandText='<%# "INSERT INTO LYInfo (ClassID, Content, Title, UserID) VALUES (?, ?, ?, ?)" %>'
ConnectionString='<%# System.Configuration.ConfigurationSettings.AppSettings("MM_CONNECTION_STRING_Cnn")
%>'
DatabaseType='<%# System.Configuration.ConfigurationSettings.AppSettings("MM_CONNECTION_DATABASETYPE_Cnn")
 %>'
Expression='<%# Request.Form("MM_insert") = "form1" %>'
CreateDataSet="false"                                    替换为Request.querystring("classid")
SuccessURL='<%# "bjly.aspx?id=" + Request.querystring("classid") %>'
Debug="true"
><Parameters>
  <Parameter Name="@ClassID" Value='<%# IIf((Request.Form("title") <> Nothing), Request.Form("title"), "")
 %>' Type="Integer" />
  <Parameter Name="@Content" Value='<%# IIf((Request.Form("Content") <> Nothing), Request.Form("Content"),
 "") %>' Type="WChar" />
  <Parameter Name="@Title" Value='<%# IIf((Request.Form("title") <> Nothing), Request.Form("title"), "")
%>' Type="WChar" />
  <Parameter Name="@UserID" Value='<%# IIf((Request.Form("title") <> Nothing), Request.Form("title"), "")
%>' Type="Integer" />
</Parameters>                                   替换为Request.Cookies("id").value
</MM:Insert>
```

图 10.73　【插入记录】对话框对应的代码

(5) 修改代码之后的页面可以正常运行。但也存在一个小问题，即当用户在【服务器行为】面板中双击【插入记录】对话框对应的项目以对其进行修改时，是无法打开【插入记录】对话框的，系统将会提示错误信息，如图 10.74 所示。

图 10.74　提示对话框

这是由于在 Dreamweaver 8 的【插入记录】对话框中，其列的获取值是无法识别非表单元素变量的。因此，当对【插入记录】对话框添加代码无须再次通过该对话框修改时，可采用修改代码的方法来解决非表单变量获取值的问题；否则，则还需通过第一种方法来解决，即在页面中添加隐藏域。这点，需要特别提示大家注意。

此外，当提交留言信息时，可能会存在所输入的中文无法保存的问题。此时，只需将切换至【代码】视图，并将页首第一行代码中的“ResponseEncoding="gb2312"”去掉即可，如图 10.75 所示。

```
<%@ Page Language="VB" ContentType="text/html" ResponseEncoding="gb2312" %>   删除
<%@ Register TagPrefix="MM" Namespace="DreamweaverCtrls" Assembly=
"DreamweaverCtrls,version=1.0.0.0,publicKeyToken=836f606ede05d46a,culture=neutral" %>
```

图 10.75　修改代码

(6) 在【代码】视图中添加页面的 Page_Load 事件代码，以判断当前用户是否为合法用户。事件代码与浏览留言中的 Page_Load 事件代码完全一致，这里不再赘述。

至此，发表留言的页面功能全部完成，页面预览效果如图 10.76 所示。

图 10.76　页面预览效果

10.5　班 级 相 册

在班级相册中，将展示本班级同学上传的相片信息，其功能包括浏览相册、查看相片和上传相片等。其中，在查看相片功能模块中，除放大查看指定的相片外，还可查看其他同学对此相片的评论以及发表自己的评论。

10.5.1　浏览相册

新建一个 ASP.NET VB 类型的页面，将其命名为 bjxc.aspx。选择【修改】|【模板】|【套用模板到页】命令，在弹出的【选择模板】对话框中，选择【页面模板】。单击【确定】按钮，将模板应用到本页面。

在可编辑区域 EditRegion1 的表单 Form 中，插入一个 1 行 2 列的表格 table2，表格属性设置如图 10.77 所示。

图 10.77　表格属性

设置第 2 个单元格的水平对齐方式为【右对齐】，并在两个单元格中分别插入图片和文本，如图 10.78 所示。

图 10.78　表格设置

图中，文本“我的同学录”的链接设置为 Main.aspx，目标为_self；文本“上传相片”的链接设置为“bjxc_Add.aspx?ClassID=<%=request.QueryString("id")%>”，目标为_self。在文本“我的同学录”后面的两个尖括号之后，将分别绑定用来显示当前班级的所属学校名称以及班级名称的字段。

1. 创建一个数据集

(1) 创建一个数据集，为动态显示学校名称与班级名称提供数据源。该数据集与浏览留言页面中的数据集 DataSet1 完全一样，这里不再介绍其创建过程。

(2) 创建数据集 DataSet1 之后，在【绑定】面板中展开数据集 DataSet1。分别选择并拖动字段 SchoolName 和 ClassName 至文本“我的同学录>”和文本“我的同学录>>”之后，并分别设置其字体颜色为红色。

(3) 在表格 table2 下，插入一个 1 行 1 列的表格 table3，其属性设置如图 10.79 所示。

图 10.79　表格属性

该表格用于为相册浏览区域添加一个边框，边框颜色值为“#0033FF”。在该表格内，仅有一个单元格，此单元格便是用于显示所要浏览的班级相册信息，该信息将通过【服务器行为】面板中的【数据列表】命令来绑定实现。

在绑定数据显示之前，还需要创建一个数据集。由于所要创建的数据集与数据集 DataSet1 具有相同的参数，因此可通过复制数据集 DataSet1 来生成。

(4) 复制数据集 DataSet1，并粘贴生成新的数据集 DataSet2。双击数据集 DataSet2，在弹出的【数据集】对话框中修改 SQL 语句为：

```
SELECT photoinfo.*,('~/photo/'+photoaddr) as photo_addr,username,(select
count(*) FROM photo_pl WHERE photoid=photoinfo.id) as pl_num FROM
photoinfo,usersheet WHERE upuser=usersheet.code and classcode =? ORDER BY
update desc
```

如图 10.80 所示。

(5) 单击【确定】按钮，完成数据集 DataSet2 的设置，该数据集将返回指定班级的所有相片信息。其中，字段 Pl_Num 为相片的评论数。

2. 创建用于显示单条相片信息的表格

(1) 在表格 table3 中创建一个用于显示单条相片信息的表格，表格设计如图 10.81 所示。

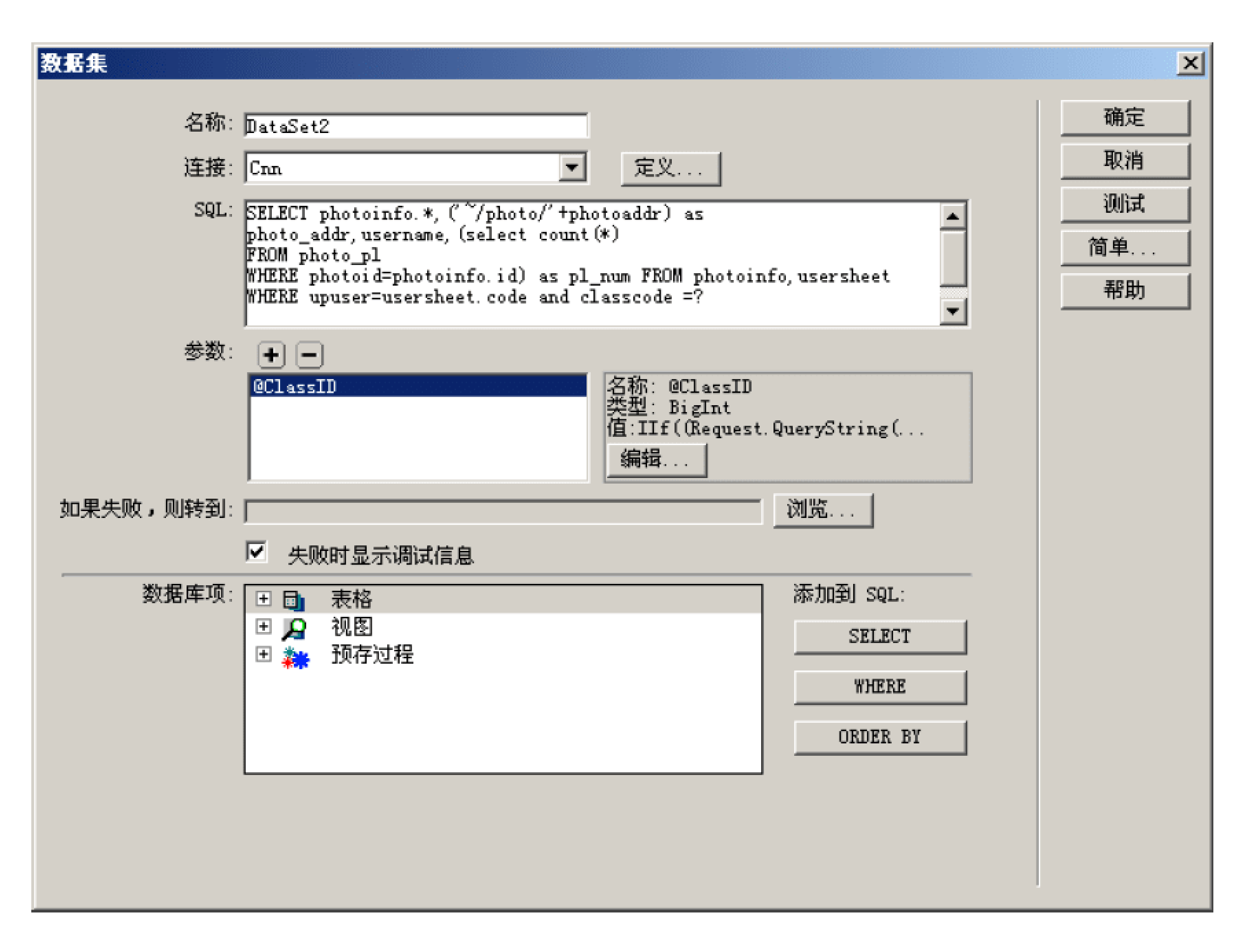


图 10.80　编辑数据集 DataSet2

图 10.81 中，浅色字体标记了各处所需要绑定哪些动态数据。此外，在文本“点击”和“评论”后将分别绑定显示相片的点击次数和评论次数。

(2) 在【绑定】面板中展开数据集 DataSet2，此时系统将会弹出错误提示对话框，如图 10.82 所示。

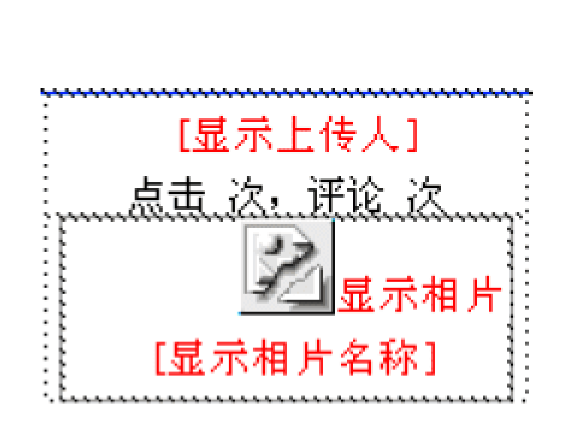


图 10.81　表格设计

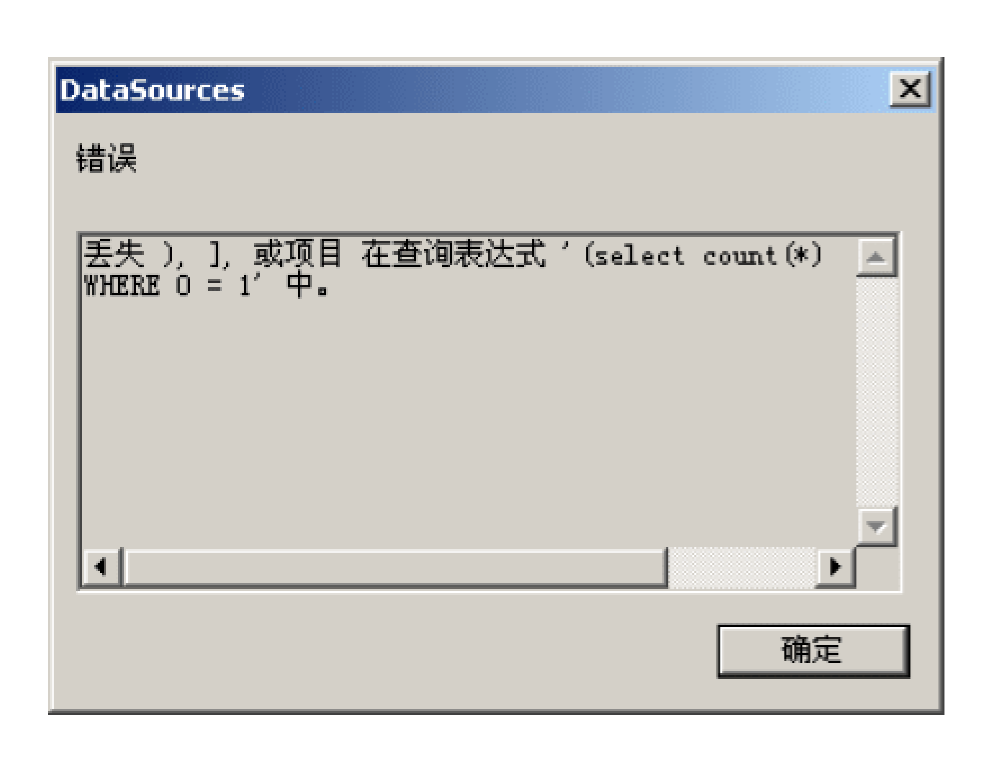
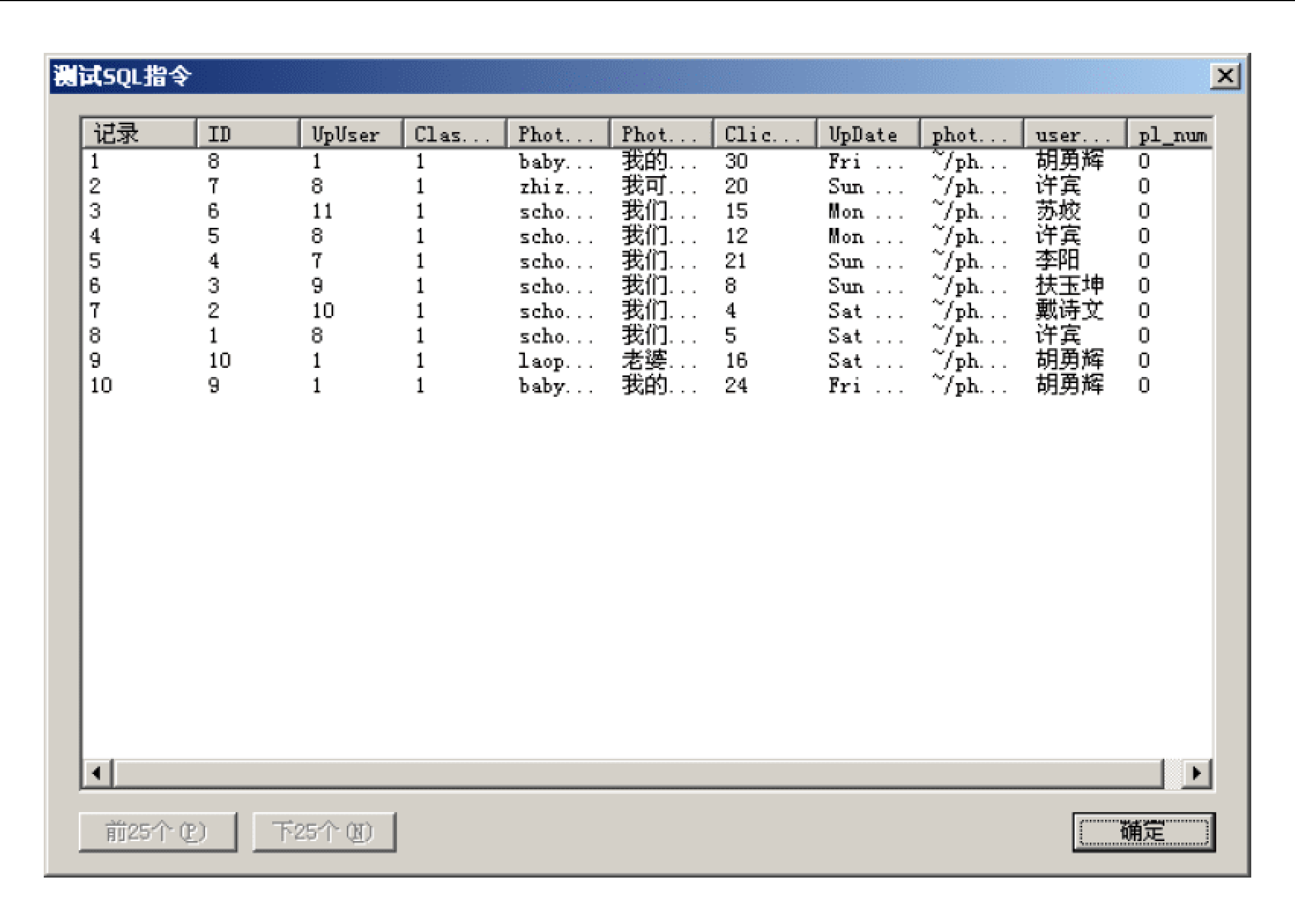


图 10.82　错误提示对话框

这并非是用户所创建的数据集有问题。事实上，在数据集 DataSet2 的设计窗口中，单击【测试】按钮，在弹出的【请提供一个测试值】对话框中，输入变量@ClassID 的测试值 1，然后单击【确定】按钮，系统将显示测试结果，如图 10.83 所示。

这说明，在数据集 DataSet2 中所定义的 SQL 语句没有问题。那么，在【绑定】面板中所弹出的错误提示对话框是怎么回事呢？这是由于数据集 DataSet2 中所定义的 SQL 语句包含了 Select 子查询语句(该语句用于返回相片的评论数)，而在 Dreamweaver 8 中并不支持对包含 Select 子句的 SQL 语句的可视化绑定操作。



记录	ID	UpUser	Clas...	Phot...	Phot...	Clic...	UpDate	phot...	user...	pl_num
1	8	1	1	baby...	我的...	30	Fri ...	~/ph...	胡勇辉	0
2	7	8	1	zhiz...	我可...	20	Sun ...	~/ph...	许宾	0
3	6	11	1	scho...	我们...	15	Mon ...	~/ph...	苏姣	0
4	5	8	1	scho...	我们...	12	Mon ...	~/ph...	许宾	0
5	4	7	1	scho...	我们...	21	Sun ...	~/ph...	李阳	0
6	3	9	1	scho...	我们...	8	Sun ...	~/ph...	扶玉坤	0
7	2	10	1	scho...	我们...	4	Sat ...	~/ph...	戴诗文	0
8	1	8	1	scho...	我们...	5	Sat ...	~/ph...	许宾	0
9	10	1	1	laop...	老婆...	16	Sat ...	~/ph...	胡勇辉	0
10	9	1	1	baby...	我的...	24	Fri ...	~/ph...	胡勇辉	0

图 10.83　测试结果

不过，经过前面的实践，相信大家对绑定字段的代码已经比较熟悉。因此，这里可以切换至【代码】视图，在需要绑定字段的地方直接输入相应的绑定代码，如图 10.84 所示。

```
        <table width="160" align="center" cellpadding="0" cellspacing="0" border="0">
        <tr height="20">
          <td align="center">
            <div style="line-height:1.2">
            <span class="STYLE2"><%# DataSet2.FieldValue("username", Container) %></span><br />
            点击<span class="STYLE1"><%# DataSet2.FieldValue("clicknum", Container) %></span>次，评论
<span class="STYLE1"><%# DataSet2.FieldValue("pl_num", Container) %></span>次
            </div>
          </td>
         </tr>
         <tr>
           <td align="center" style="padding-top:5px;">
             <table width="150" cellpadding="0" cellspacing="1" border="0" bgcolor="#999999">
             <tr>
               <td bgcolor="#FFFFFF" align="center" style="padding-top:5px;padding-bottom:2px;">
<asp:Image ID="Image1" Height=100 ImageUrl=<%# DataSet2.FieldValue("photo_addr", Container) %> runat=
"server" /><br /><%# DataSet2.FieldValue("photoname", Container) %> </td>
             </tr>
             </table>
           </td>
         </tr>
         </table>
```

相片上传人
点击次数
评论次数
相片名称
相片路径

图 10.84　添加绑定字段代码

图 10.84 中，用文字标记显示了所要添加绑定字段的地方，并说明了各绑定字段的含义。事实上，在 Dreamweaver 8 的可视化操作中，尚存在一些不足之处(特别是对某些 ASP.NET 服务器控件和数据集的处理)。此时，完全可以直接编辑代码，来实现所需要的功能。

(3) 添加对上传人和相片的超级链接。其中，上传人的链接地址为“bjtxl.aspx?id=<%# DataSet2.FieldValue("classcode", Container) %>#<%# DataSet2.FieldValue("upuser", Container) %>”，相片的链接地址为“viewphoto.aspx?id=<%# DataSet2.FieldValue("id", Container) %>”。

(4) 在完成绑定字段的设置后，在【代码】视图中选择并剪切整个表格的定义代码。

在【服务器行为】面板中单击＋按钮，从弹出的菜单中选择【数据列表】命令。在弹出的【数据列表】对话框中，设置数据集为 DataSet2，并设置显示所有记录。在【模板】列表框中选择【项目】，并在【内容】文本框中粘贴刚才所剪切的代码。同时，设置【组织项】为【使用表】，【表列】为 4，如图 10.85 所示。

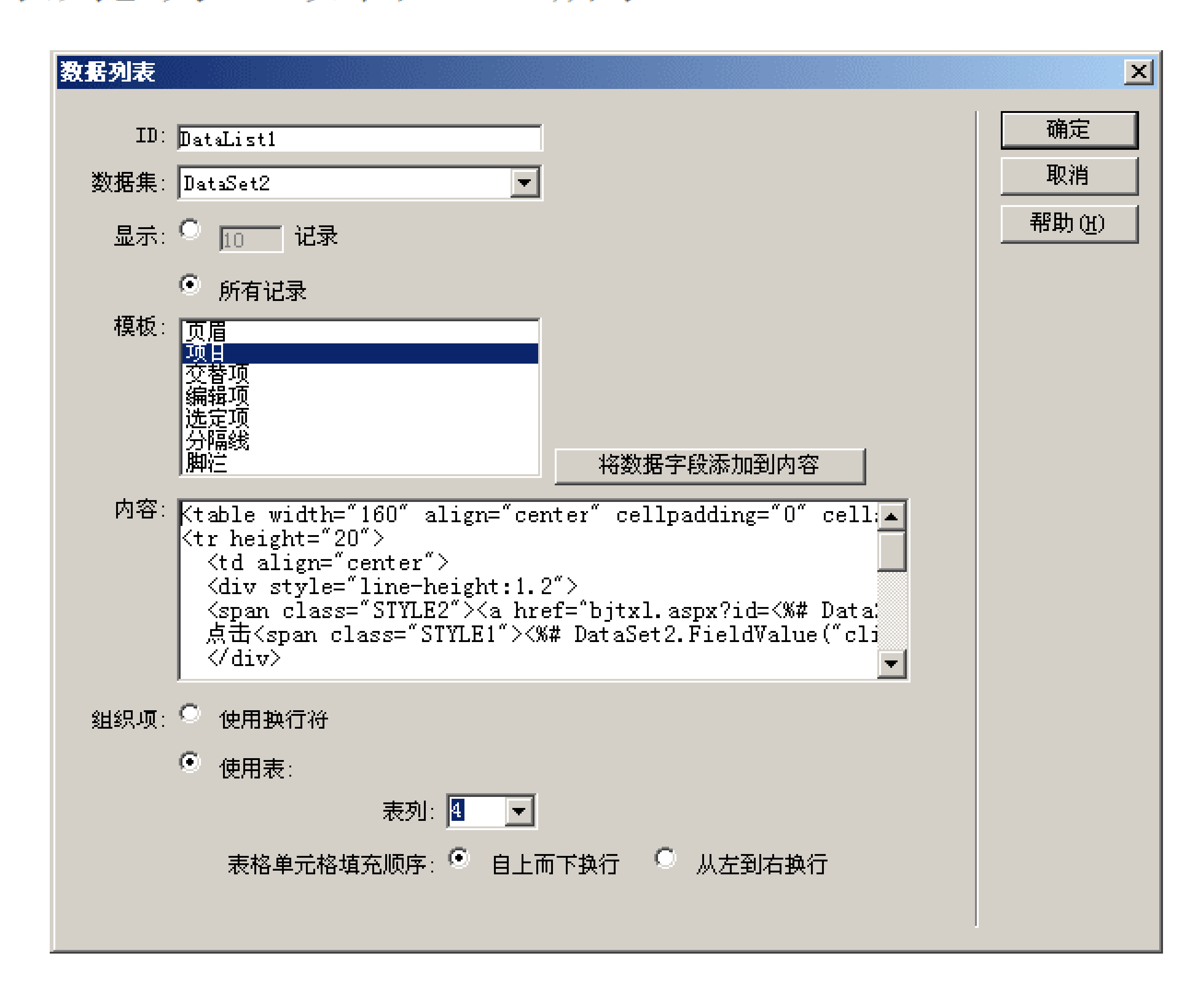


图 10.85　创建数据列表

(5)　单击【确定】按钮，完成数据列表的创建。

至此，浏览相册的功能全部完成，页面预览效果如图 10.86 所示。

图 10.86　页面预览效果

10.5.2　查看相片

在浏览相册时，可单击具体的相片，打开一个新的页面以查看更大尺寸的相片，该页还显示了其他同学对该相片的相关评论，并且在该页还可发表自己的评论。

新建一个 ASP.NET VB 动态页，将其命名为 ViewPhoto.aspx。选择【修改】|【模板】|【套用模板到页】命令，在弹出的【选择模板】对话框中选择【页面模板】。单击【确定】按钮，将模板应用到本页面。

复制浏览相册中的表格 table2，粘贴到可编辑区域 EditRegion1 中的表单 Form 中，如图 10.87 所示。

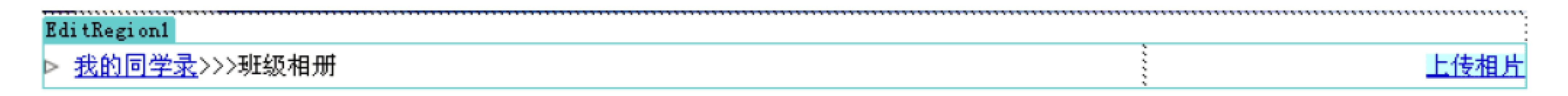


图 10.87　复制表格

在表格 table2 的下面，再插入一个 1 行 4 列的表格 table3，其表格属性设置如图 10.88 所示。

图 10.88　表格属性

该表格用于显示所查看相片的基本信息，包括相片名称、上传人、点击次数以及上传时间等。在表格的各单元格中，输入相应的文本信息，如图 10.89 所示。

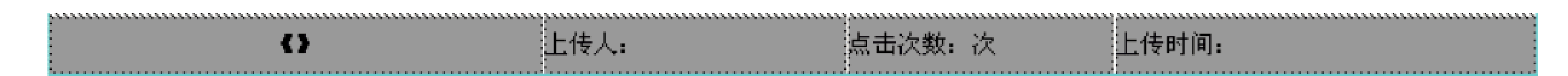


图 10.89　新插入的表格 table3

设置单元格中各字体的颜色为白色，并在表格 table3 下插入一个 Image 控件，设置其 BorderWidth 属性为 10，BorderColor 属性为“#999999”，暂不设置其 ImageUrl 属性。

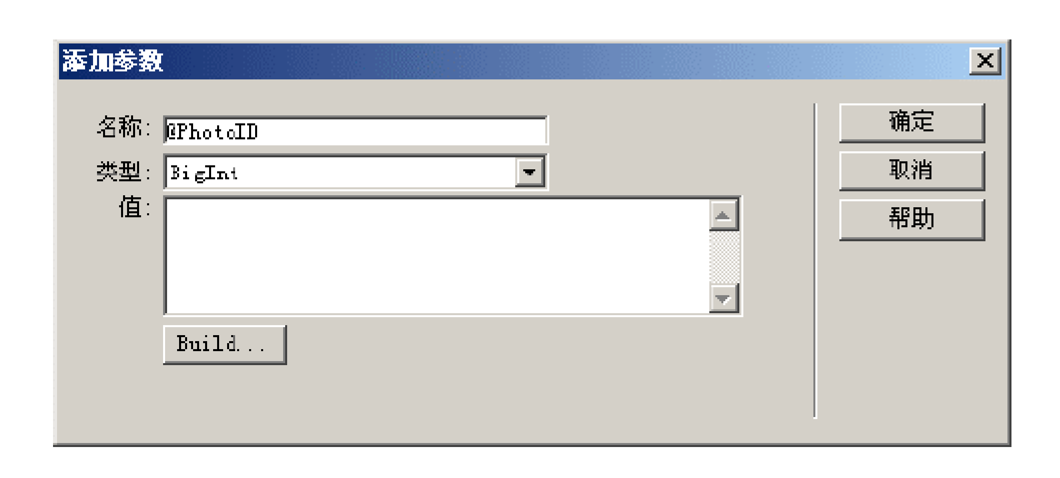
1. 创建一个数据集

下面创建一个数据集，为动态显示相片信息提供数据源。

(1) 打开【应用程序】面板组，切换至【服务器行为】面板，单击 按钮，从弹出的菜单中选择【数据集】命令。在弹出的【数据集】对话框中单击【高级】按钮，切换至【数据集】对话框的高级模式。

(2) 在【数据集】对话框的高级模式中，单击【添加参数】按钮 ，在弹出的【添加参数】对话框中设置参数名称为“@PhotoID”，如图 10.90 所示。

(3) 单击 Build 按钮，在弹出的【生成值】对话框中，设置参数的值为 URL 参数 ID，如图 10.91 所示。



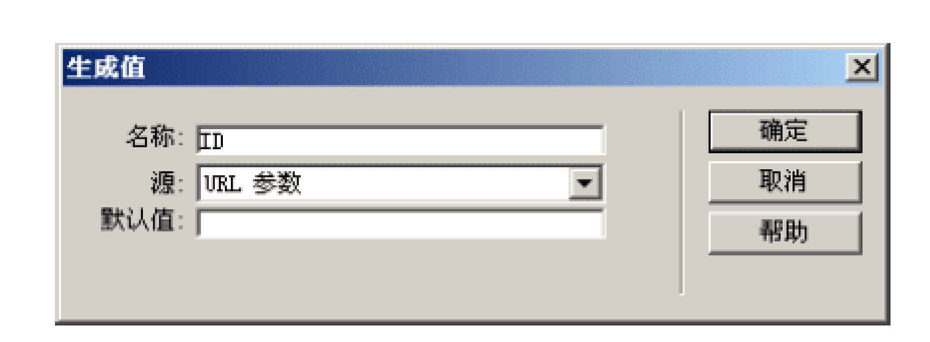


图 10.91　【生成值】对话框

图 10.90　【添加参数】对话框

(4) 单击【确定】按钮，返回【添加参数】对话框；再次单击【确定】按钮，返回【数据集】对话框。设置数据集的名称为 DataSet1，连接选择 Cnn，并在 SQL 文本框中输入以下 SQL 语句：

```
SELECT photoinfo.id,photoname,('~/photo/'+photoaddr) as
photo_addr,username,clicknum,update,classcode,classname,schoolname FROM
photoinfo,usersheet,classinfo,schoolinfo WHERE upuser=usersheet.code and
classcode=classinfo.id and schoolid=schoolinfo.id and photoinfo.id=?
```

如图 10.92 所示。

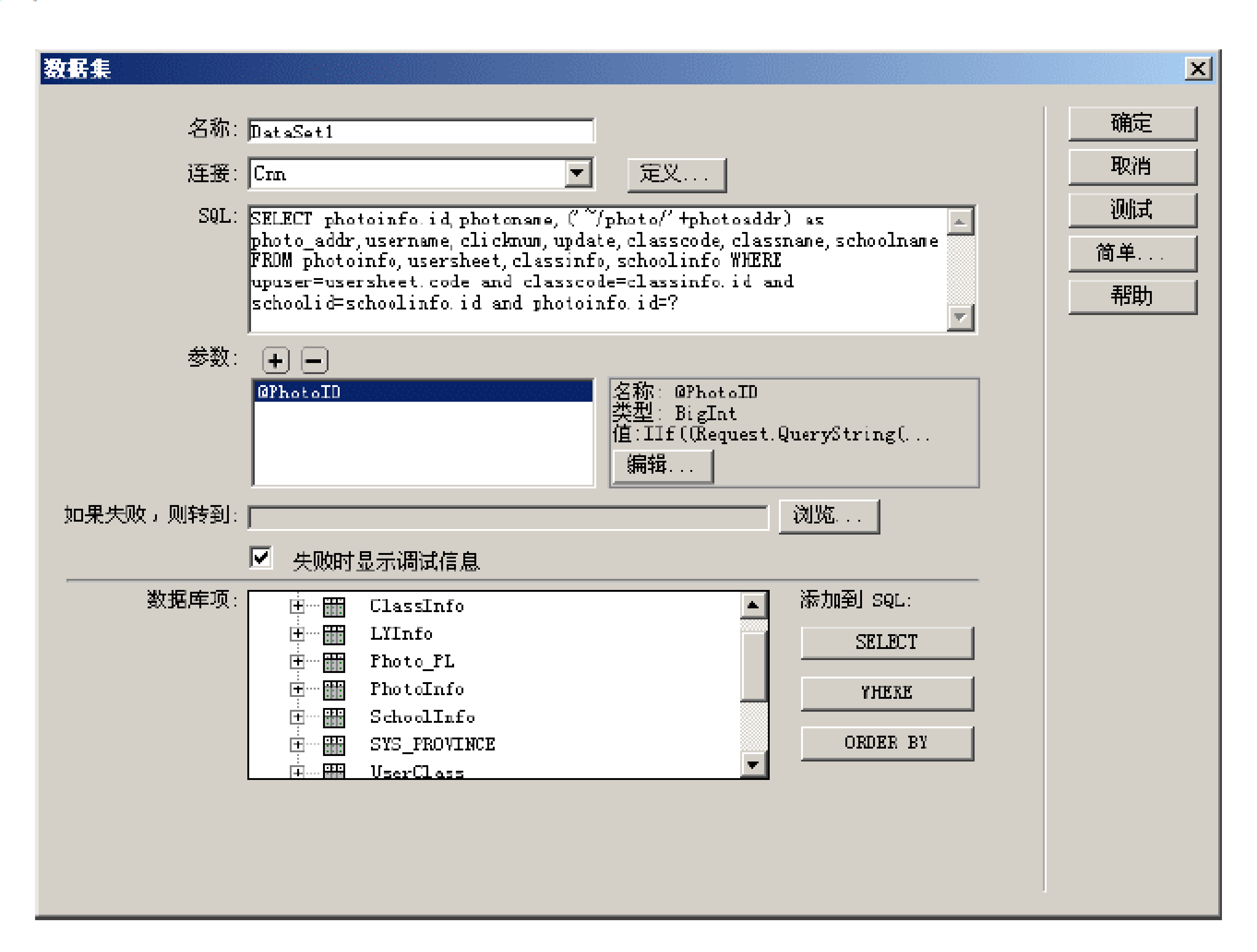


图 10.92　创建数据集 DataSet1

(5) 单击【确定】按钮，完成数据集的创建，该数据集将返回指定相片的相关信息。在【应用程序】面板组中，切换至【绑定】面板，选择并展开数据集 DataSet1 (如图 10.93 所示)，按照下面的操作绑定数据。

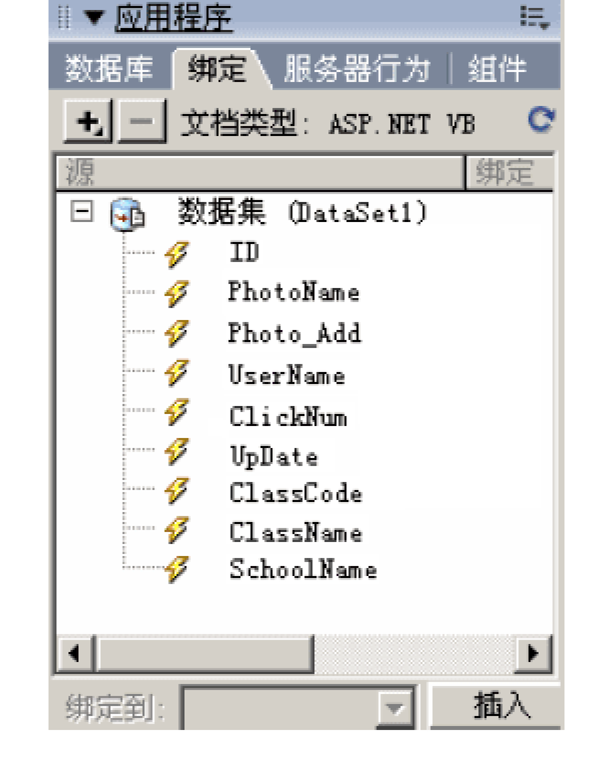


图 10.93　【绑定】面板

首先，分别选择并拖动字段 SchoolName、ClassName 至文本“我的同学录>”和“我的同学录>>”之后，并分别设置其字体颜色为红色；然后，选择并拖动字段 PhotoName 至表格 Table3 的第一个单元格中；最后，再分别选择并拖动字段 UserName、ClickNum 和 UpDate 至文本“上传人：”、“点击次数：”、“上传时间：”之后。分别选择以上动态文本，在【属性】面板中设置其为加粗显示。同时，设置 Image 控件的 ImageUrl 属性为“<%# DataSet1.FieldValue("photo_addr", Container) %>”。

此时，相片基本信息的绑定工作已完成。

2. 创建第二个数据集

下面再来创建一个数据集，为动态显示相片评论提供数据源。

(1) 在【服务器行为】面板中选择数据集 DataSet1，将其复制并粘贴以生成新的数据集 DataSet2。双击数据集 DataSet2，在弹出的【数据集】对话框中，将 SQL 语句修改如下：

```
SELECT photo_pl.*,username FROM photo_Pl,usersheet WHERE pl_user=usersheet.
code and photoid=? order by pl_date desc
```

如图 10.94 所示。

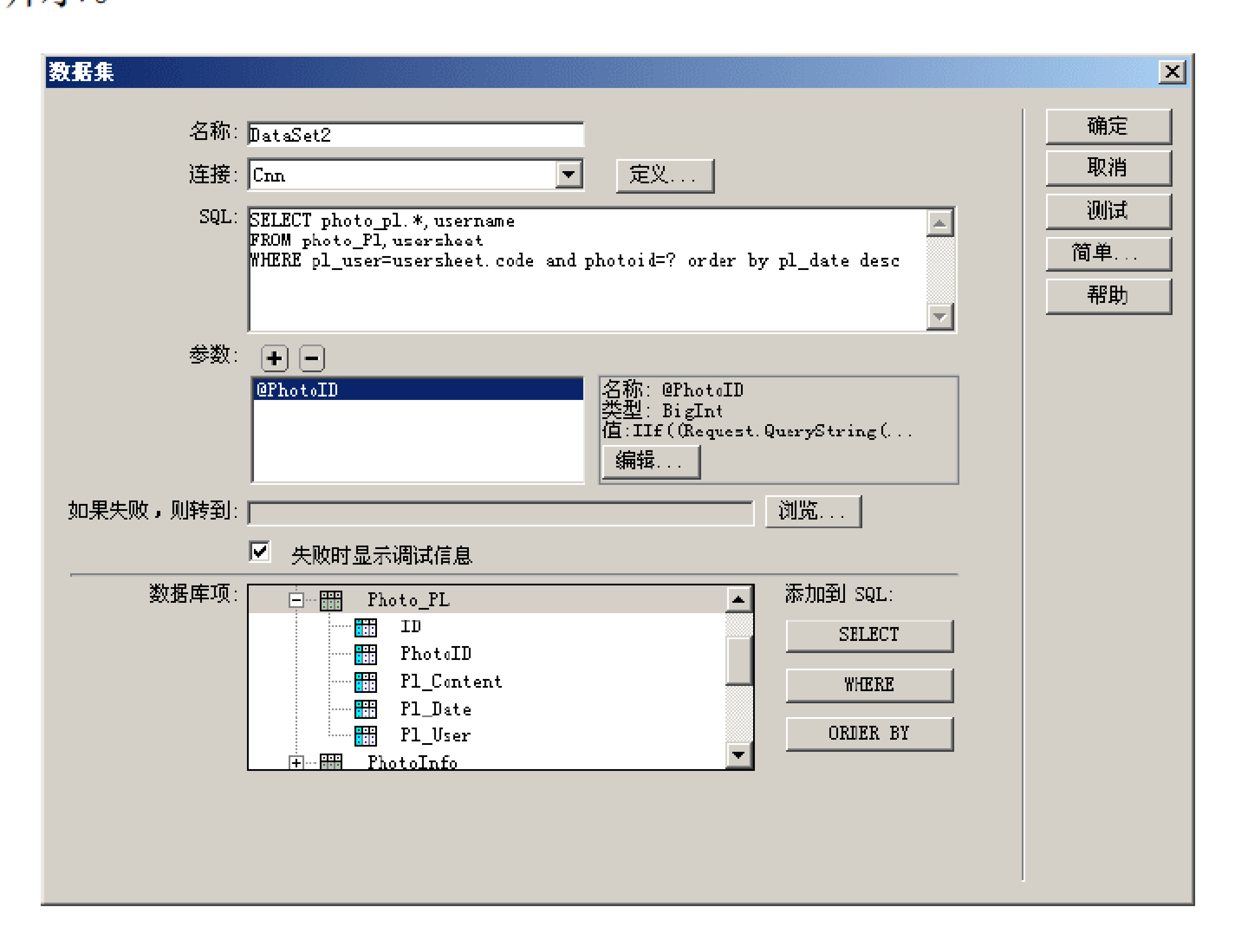


图 10.94　编辑数据集 DataSet2

单击【确定】按钮，完成对数据集 DataSet2 的设置，该数据集将返回指定相片的所有评论信息。

(2) 在 Image 控件的下方，再插入一个 3 行 1 列的表格，用于显示单条评论信息。依次设置表格中 3 个单元格的高度为 25、60 和 25，同时设置第 1 个单元格的背景色值为“#CCCCCC”，第 3 个单元格的水平对齐方式为右对齐；然后，在第 1 个和第 3 个单元格中输入相应的文本，如图 10.95 所示。

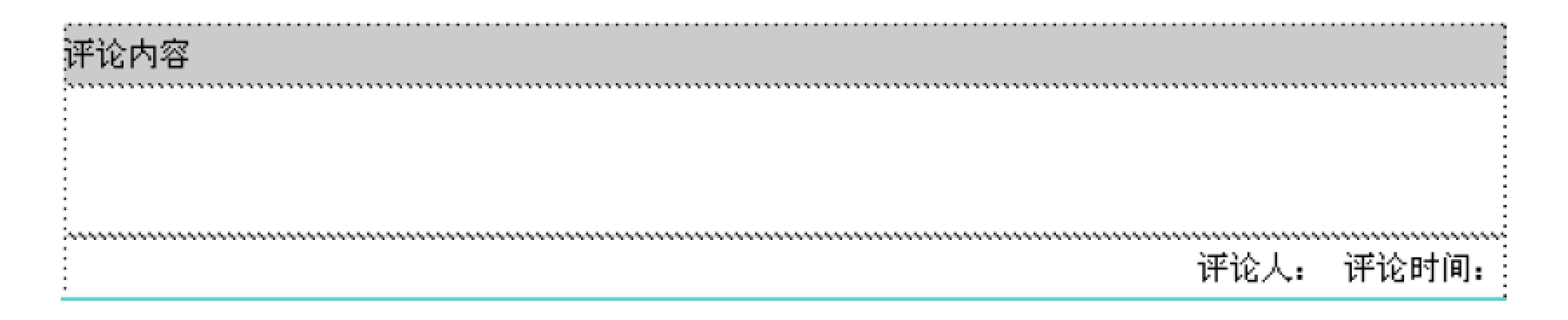


图 10.95　设置表格

(3) 在【应用程序】面板组中，切换至【绑定】面板，选择并展开数据集 DataSet2，如图 10.96 所示。

(4) 选择并拖动字段 Pl_Content 至表格的第 2 个单元格中。然后，分别选择并拖动字段 UserName、Pl_Date 至文本“评论人：”和“评论时间：”之后，并设置两个文本的字体颜色值为“#0000FF”。

(5) 在【服务器行为】面板中，选择并双击动态文本 DataSet2. Pl_Content，在弹出的【动态文本】对话框中，设置其格式为【编码–HTML 编码格式】，如图 10.97 所示。

图 10.96　【绑定】面板

图 10.97　设置动态文本格式

(6) 单击【确定】按钮，完成动态文本的设置，同时也完成了对单条评论信息的绑定。

(7) 选择整个表格，在【服务器行为】面板中单击+按钮，从弹出的菜单中选择【重复区域】命令。在弹出的【重复区域】对话框中，选择数据集 DataSet2，并设置显示所有记录，如图 10.98 所示。

单击【确定】按钮，完成重复区域的设置。

此时，对相片评论信息的数据绑定工作全部完成。还需要添加对相片评论的发表。

(8) 在前面所创建的重复区域下面插入一个 3 行 1 列的表格，并在第 1 个单元格中输入文本“请发表您对本相片的评论”，在第 2 个单元格中插入一个文本框控件，设置其 ID 为 PlInfo，文本模式为多行，行数为 6，列数为 60，如图 10.99 所示。

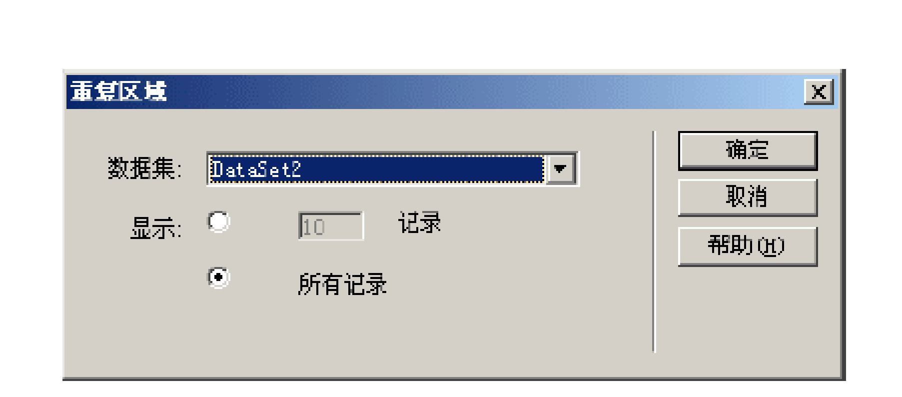


图 10.98　重复区域

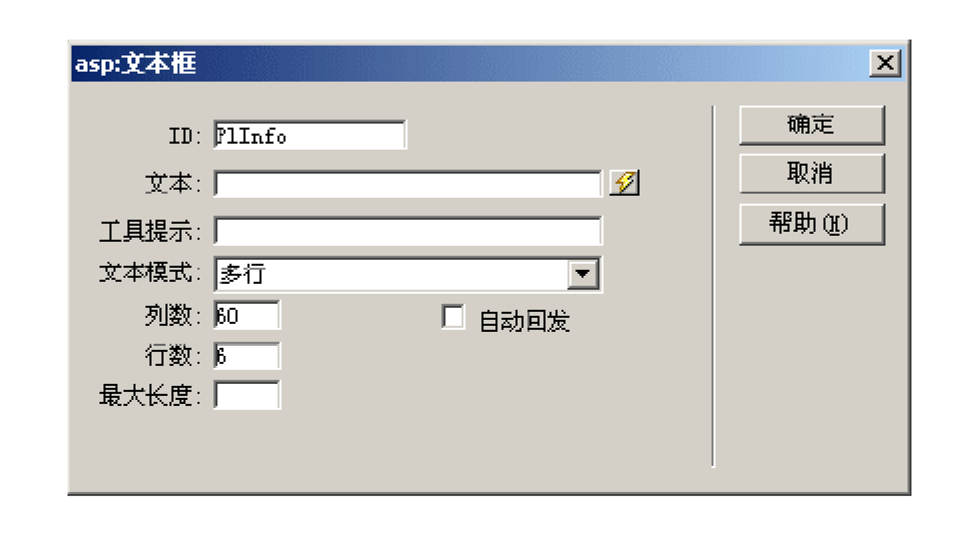


图 10.99　设置文本框控件的属性

在表格的第 3 个单元格中，插入一个提交按钮控件，此时的表格如图 10.100 所示。

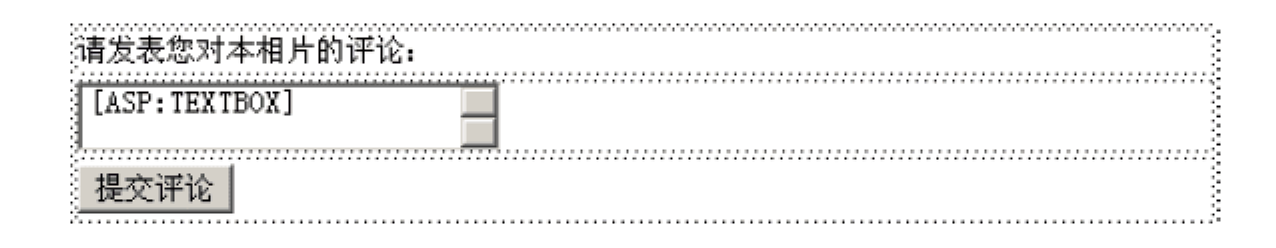


图 10.100　表格设计

(9) 实现评论信息的提交。在【服务器行为】面板中单击+按钮，从弹出的菜单中选择【插入记录】命令。在【插入记录】对话框中，连接选择 Cnn，所要插入的表格选择 Photo_PL，提交成功后所转向的页面设置为“viewphoto.aspx?id=<%# Request.querystring("id")%>”。

(10) 设置各列对应的获取值。这里需要设置的列包括 PhotoID、Pl_Content 和 Pl_User。其中，Pl_Content 列的获取值设置为表单变量 plinfo 的值。而 PhotoID 和 Pl_User 两列需要分别获取页面参数 ID 和 Cookie 变量 ID 的值。这里，通过修改代码的方法来实现。为了便

于修改代码，将 PhotoID 和 Pl_User 两列的获取值均设置为表单变量 plinfo，如图 10.101 所示。

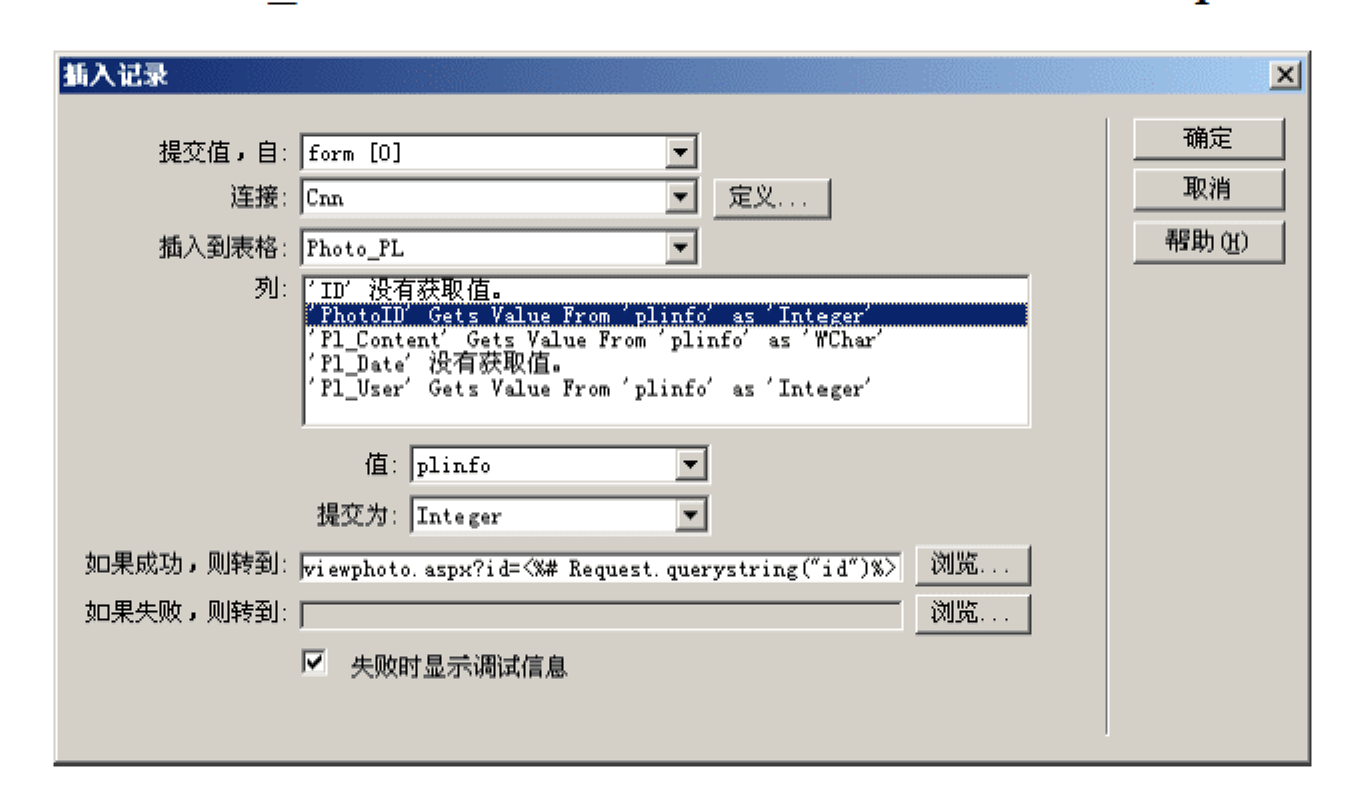


图 10.101　【插入记录】对话框

(11) 单击【确定】按钮，完成插入记录代码的添加。然后，切换至【代码】视图，找到【插入记录】服务器行为对应的代码，如图 10.102 所示。

```
<MM:Insert
runat="server"
CommandText='<%# "INSERT INTO Photo_PL (PhotoID, Pl_Content, Pl_User) VALUES (?, ?, ?)" %>'
ConnectionString='<%# System.Configuration.ConfigurationSettings.AppSettings("MM_CONNECTION_STRING_Cnn")
%>'
DatabaseType='<%# System.Configuration.ConfigurationSettings.AppSettings("MM_CONNECTION_DATABASETYPE_Cnn"
) %>'
Expression='<%# Request.Form("MM_insert") = "form1" %>'
CreateDataSet="false"
SuccessURL='<%# "viewphoto.aspx?id=" + Request.querystring("id") %>'
Debug="true"
>                                          替换成Request.Querystring("id")
  <Parameters>
    <Parameter Name="@PhotoID" Value='<%# IIf((Request.Form("plinfo") <> Nothing), Request.Form("plinfo")
, "") %>' Type="Integer" />
    <Parameter Name="@Pl_Content" Value='<%# IIf((Request.Form("plinfo") <> Nothing), Request.Form(
"plinfo"), "") %>' Type="WChar" />
    <Parameter Name="@Pl_User" Value='<%# IIf((Request.Form("plinfo") <> Nothing), Request.Form("plinfo")
, "") %>' Type="Integer" />
  </Parameters>                            替换成Request.Cookies("id").Value
</MM:Insert>
```

图 10.102　【插入记录】服务器行为对应的代码

修改变量“@PhotoID”的取值，将表达式“Request.Form("plinfo")”替换成“Request.Querystring("id")”；然后修改变量“@Pl_User”的取值，将表达式“Request.Form("plinfo")”替换成“Request.Cookies("id").Value】。此外，别忘了将页首第一句话中的“ResponseEncoding="gb2312"”去掉。否则，中文输入将不会被认可。

(12) 在【代码】视图添加页面的 Page_Load 事件。在该事件中，除了需要对用户的合法身份进行验证外，还需在数据库中对指定相片的点击次数进行加 1 操作。

【示例代码】

```
Sub Page_Load(ByVal Sender As Object, ByVal E As EventArgs)
  Dim StrCnn As String
  Dim StrSql As String
  Dim Cnn As OleDbConnection
  Dim Cmd As OleDbCommand
  If Not IsPostBack() Then
    If IsDBNull(Request.Cookies("id").Value) Or
Trim(Request.Cookies("id").Value) = "" Then
      Response.Redirect("default.aspx")  '将页面跳转至登录页面
```

```
    else
      if request.QueryString("id")<>"" then
        StrCnn =
System.Configuration.ConfigurationSettings.AppSettings("MM_CONNECTION_ST
RING_Cnn")  '获取数据库连接字符串
        Cnn = New OleDbConnection(StrCnn)
        Cnn.Open()  '打开数据库连接
        StrSql = "update photoinfo set clicknum=clicknum+1 where id=" &
request.QueryString("id")  '对指定相片的点击次数执行加 1 操作
        Cmd = New OleDbCommand(StrSql, Cnn)
        Cmd.ExecuteNonQuery()
        Cnn.Close()  '关闭数据库连接
      end if
    End If
  End If
End Sub
```

至此，查看相片的功能页面制作完成，页面预览效果如图 10.103 所示。

图 10.103　页面预览效果

10.5.3　上传相片

上传相片功能具体是指添加新的相片至指定班级的班级相册中。

1. 页面基本结构的设计

新建一个 ASP.NET VB 类型的页面，将其命名为 Bjxc_Add.aspx。选择【修改】|【模板】|【套用模板到页】命令，在弹出的【选择模板】对话框中，选择【页面模板】。单击【确定】按钮，将模板应用到本页面。

上传相片页面与发表留言页面布局上较为相似，因此可直接复制发表留言页面的

HTML 代码，粘贴上传相片页面，并在其基础上进行修改。

打开发表留言页面(Bjly_Add.aspx)，将视图切换至【代码】视图，并将表单 Form 之内的所有 HTML 代码(不含【插入记录】服务器行为生成的隐藏域)复制到上传相片页面(Bjxc_Add.aspx)中的表单 Form 内，并在【设计】视图中对其进行调整，调整后的布局如图 10.104 所示。

图 10.104　调整后的布局

在文本“选择相片”后的单元格中，添加了一个 FileUpload 控件，该控件是 ASP.NET 2.0 中新增的控件，Dreamweaver 8 尚不支持对它的可视化操作。因此，只能在【代码】视图编写该控件的定义代码，如下：

```
<asp:FileUpload ID="FileUpload1" runat="server" ToolTip="选择相片"
Width="420px" EnableTheming="True" />
```

FileUpload 控件将在界面上显示为一个文件框控件和一个【浏览】按钮控件，主要用于实现文件的上传。

在【提交】按钮控件下方，添加了一个 Label 控件和一个 Image 控件。前者用于显示上传文件时可能出现的错误信息，而后者则用于当单击【预览】按钮时，显示用户所选择的上传文件的图像。

2. 创建数据集

首先，创建一个数据集，用来实现动态显示当前指定班级所属的学校名称以及班级名称。

(1) 打开【服务器行为】面板，单击按钮，从弹出的菜单中选择【数据集】命令。在弹出的【数据集】对话框中单击【高级】按钮，切换至【数据集】对话框的高级模式。

(2) 在【数据集】对话框的高级模式中，单击【添加参数】按钮，在弹出的【添加参数】对话框中设置参数名称为“@ClassID”，设置其值为 URL 参数 ClassID。然后，在 SQL 文本框中输入以下 SQL 语句：

```
select classinfo.id,classname,schoolname from classinfo,schoolinfo where
schoolid=schoolinfo.id and classinfo.id=?
```

如图 10.105 所示。

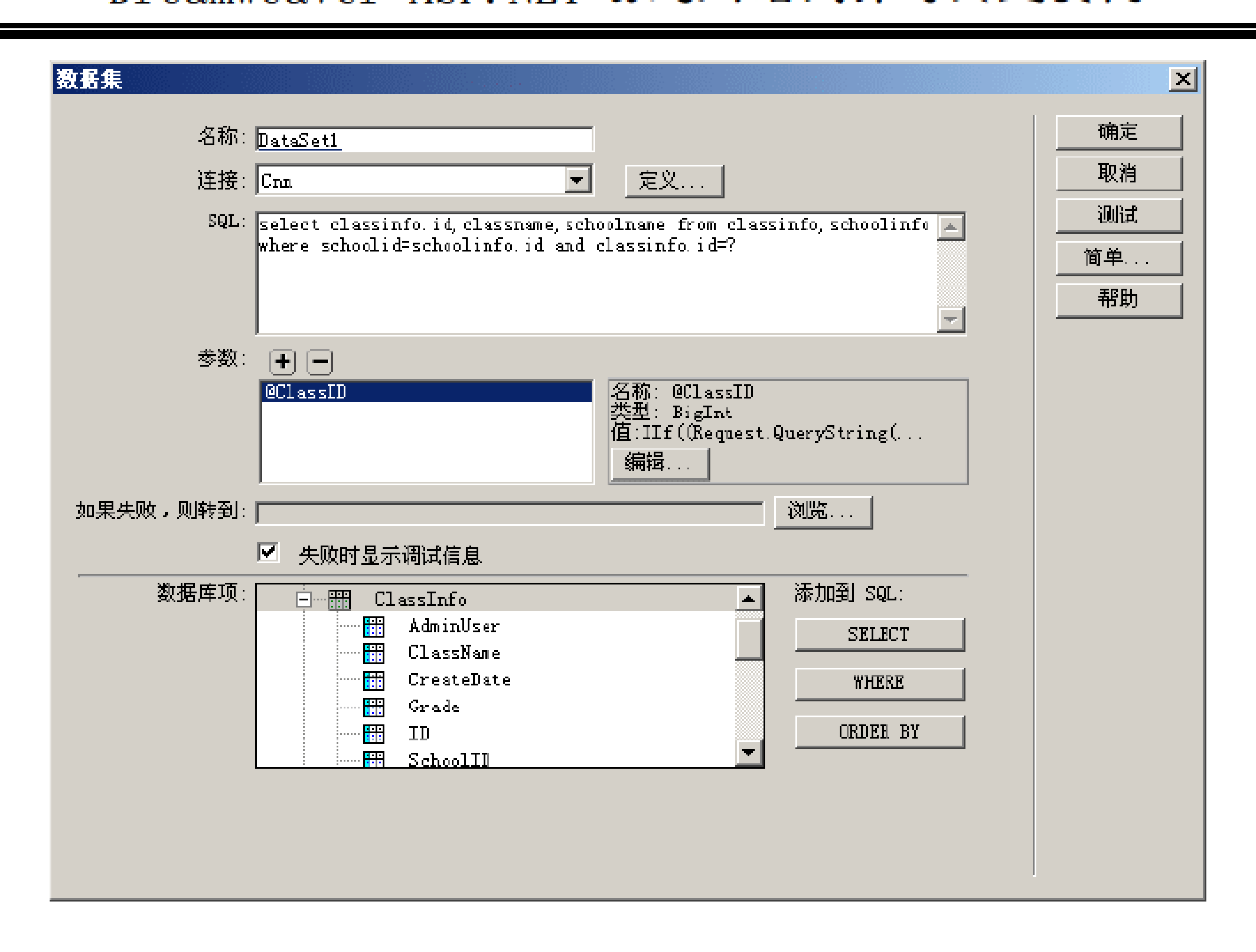


图 10.105　创建数据集 DataSet1

(3) 单击【确定】按钮，完成数据集 DataSet1 的创建。在【应用程序】面板组中，切换至【绑定】面板，选择并展开数据集 DataSet1，如图 10.106 所示。

(4) 分别选择并拖动字段 SchoolName、ClassName 至文本“我的同学录>”和“我的同学录>>”之后，并设置其字体颜色为红色。

(5) 在页面上，添加了一个【预览】按钮，该按钮并非 ASP.NET 中的 Button 服务器控件，而是 HTML 中的<Input type="button">标签。当单击【预览】按钮时，将执行与其关联的 JavaScript 脚本函数 DispPhoto()，该函数的定义如下：

图 10.106　创建数据集 DataSet1

【示例代码】

```
Function dispphoto(){
  if
(document.getElementById("FileUpload1").value!="")
  {
   //设置预览图像的显示宽度

   document.getElementById("Image1").style.width="300px";
   //获取上传文件输入框的值，并将其赋予 Image 控件的 scr 属性

   document.getElementById("Image1").src=document.getElementById("
FileUpload1").value;
  }
}
```

(6) 对于【提交】按钮，所执行的操作包括两步：①上传用户指定的文件；②将上传的相片信息保存至数据库中。

单击【提交】按钮，所执行的事件代码如下：

【示例代码】

```
Sub Save_Click(ByVal Sender As Object, ByVal E As EventArgs)
  Dim Sql As String
  Dim NameStr As String = title.Text  '获取相片名称信息
  Dim UserStr As String=Request.Cookies("ID").value  '获取当前用户 ID
  Dim ClassStr AS String=Request.QueryString("ClassID") '获取班级 ID
  Dim PathStr As String
  Dim ErrStr As String
  Dim Cnn As OleDbConnection
  Dim Cmd As OleDbCommand
  Dim StrCnn As String
  '调用 FileUpload 控件的 HasFile 方法判断用户是否选择了相片文件
  If FileUpload1.HasFile Then
    '用户选择了文件，开始上传
    Try
      '设置文件上传的目标路径
      PathStr = Server.MapPath("data.mdb")
      PathStr = Left(PathStr, InStrRev(Trim(PathStr), "\") - 1)
      PathStr = PathStr & "\photo\" & FileUpload1.FileName
      FileUpload1.SaveAs(PathStr)  '执行上传操作
    Catch ex As Exception
      '上传文件失败，提示错误信息
      ErrStr = "相片上传失败：" & Replace(ex.Message.ToString, "/", "//")
      Label1.Text = Trim(ErrStr)
      Exit Sub '退出过程
    End Try
    '获取数据库连接字符串
    StrCnn = System.Configuration.ConfigurationSettings.AppSettings
("MM_CONNECTION_STRING_Cnn")
    Cnn = New OleDbConnection(StrCnn)
    Cnn.Open()
    '向数据表 PhotoInfo 中插入所上传的相片信息
    Sql = "insert into photoinfo(upuser,classcode,photoaddr,photoname)
values(" & trim(UserStr) & "," & trim(ClassStr) & ",'" &
Trim(FileUpload1.FileName) & "','" & Trim(NameStr) & "')"
    Cmd = New OleDbCommand(Sql, Cnn)
    Cmd.ExecuteNonQuery()
    Cnn.Close()
    '数据保存后，将页面跳转至班级相册页面
    Response.Redirect("bjxc.aspx?id=" & request.QueryString("classid"))
  End If
End Sub
```

至此，实现了上传相片页面的功能，页面预览效果如图 10.107 所示。

选择要上传的相片文件后，单击【预览】按钮，可在下方预览该图片，如图 10.108 所示。

图 10.107　页面预览效果

图 10.108　相片预览

10.6　班级通讯录

在班级通讯录页面可以查看指定班级中所有同学的联系方式，其数据来源主要从用户的注册信息中获取。

新建一个 ASP.NET VB 类型的页面，将其命名为 Bjtxl.aspx。选择【修改】|【模板】|【套用模板到页】命令，在弹出的【选择模板】对话框中选择【页面模板】。单击【确定】

按钮，将模板应用到本页面。

与前面页面类似，首先在表单 Form 中插入一个 1 行 1 列的表格，用于显示当前页面所在的班级，如图 10.109 所示。

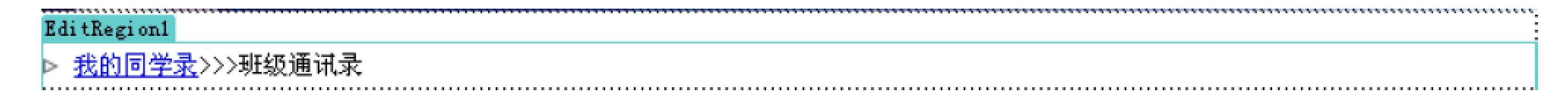


图 10.109 插入的表格

1. 创建第一个数据集

创建数据集 DataSet1，用于实现动态显示当前班级所属的学校名称以及班级名称。该数据集与上传相片页面中所定义的数据集 DataSet1 一样，仅参数的取值不同。前者为 URL 参数 ID，而后者为 URL 参数 ClassID。对于数据集 DataSet1 的创建过程，这里不再赘述。

在【应用程序】面板组中，切换至【绑定】面板，选择并展开数据集 DataSet1。然后分别选择并拖动字段 SchoolName、ClassName 至文本“我的同学录>”和“我的同学录>>”之后，并设置其字体颜色为红色。

接下来，在该表格下插入一个 3 行 3 列的表格 table3，其属性设置如图 10.110 所示。

图 10.110 表格属性

该表格用于显示指定班级的同学列表。分别合并该表格第 1 行和第 3 行的 3 个单元格，并分别设置其单元格的背景图片为 Image 目录下的 txl_1.gif 和 txl_3.gif。然后，分别设置第 2 行的第 1 个单元格和第 3 个单元格的背景图片均为 Image 目录下的 txl_2.gif。最后，在第 2 行的第 2 个单元中输入文本“同学列表”，并设置其字体颜色为红色。除第 2 行的第 2 个单元格用于显示数据外，其余单元格均用于控制表格边框的显示。

2. 创建第 2 个数据集

再创建一个新的数据集，用于实现同学列表信息的动态显示。

(1) 在【服务器行为】面板中，复制数据集 DataSet1 并粘贴，以生成新的数据集 DataSet2。双击数据集 DataSet2，在弹出的【数据集】对话框中，将 SQL 语句修改如下：

```
SELECT usersheet.* FROM usersheet,userclass WHERE usercode=usersheet.code
and classcode=? ORDER BY usersheet.code
```

如图 10.111 所示。

单击【确定】按钮，完成数据集 DataSet2 的设置，该数据集将返回指定班级中的所有同学信息。

(2) 在【应用程序】面板组中，切换至【绑定】面板，选择并展开数据集 DataSet2，如图 10.112 所示。

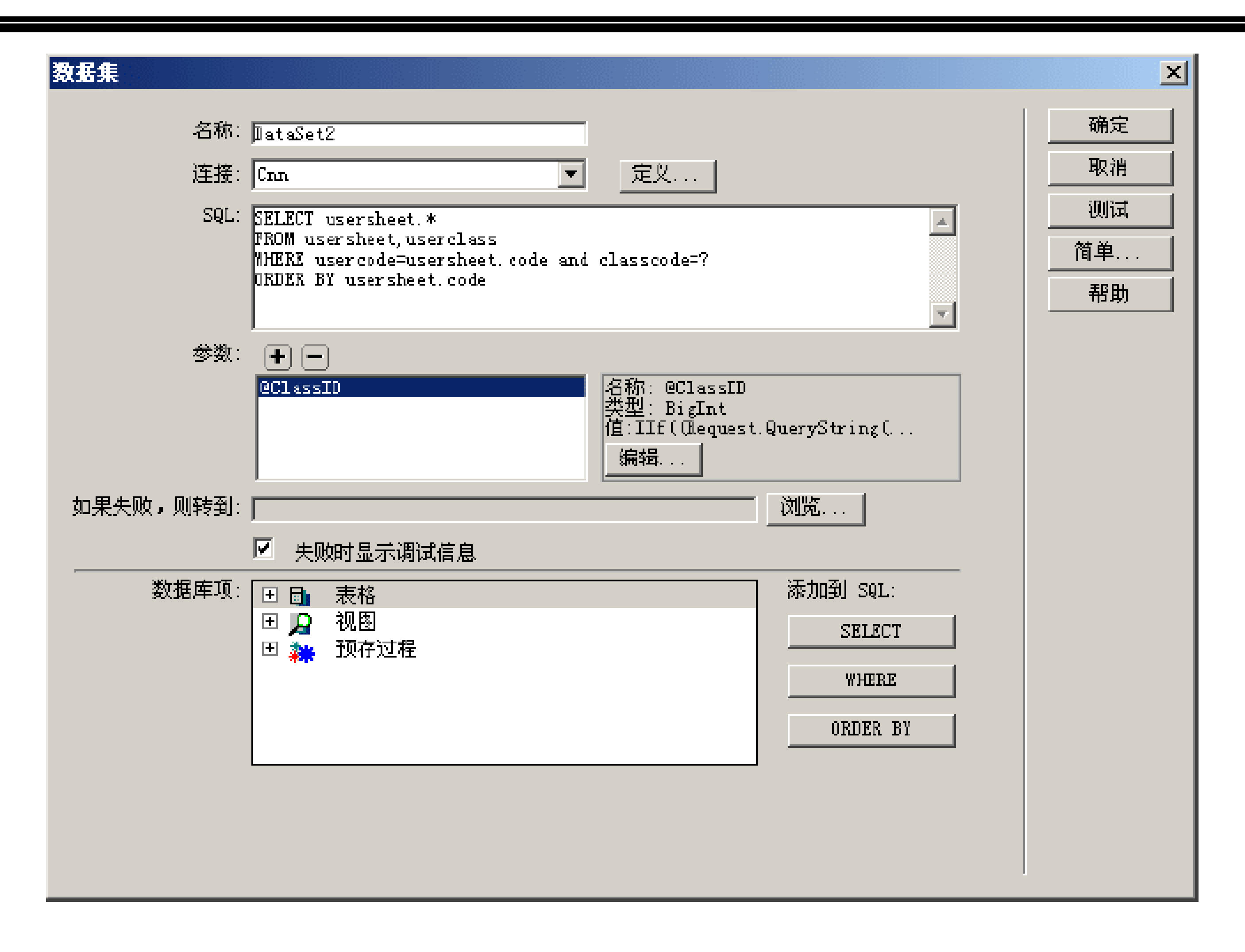


图 10.111　编辑数据集 DataSet2

选择并拖动字段 UserName 至表格 table3 中的文本“同学列表”的下面，并在【属性】面板中设置其链接为“#<%# DataSet2.FieldValue("Code", Container) %>”，然后在该文本之前插入一个图片，其文件为 Image 目录下的 arrow.gif。

(3) 在【代码】视图中，选择并剪切以上所插入的动态文本及图片对应的代码。在【应用程序】面板组中，切换至【服务器行为】面板，单击[+]按钮，在弹出的菜单中选择【数据列表】命令。在弹出的【数据列表】对话框中，设置数据集为 DataSet2，并设置显示所有记录。在【模板】列表中选择【项目】，并在【内容】文本框中粘贴刚才剪切的代码，设置【组织项】为【使用表】，【表列】为 5，如图 10.113 所示。

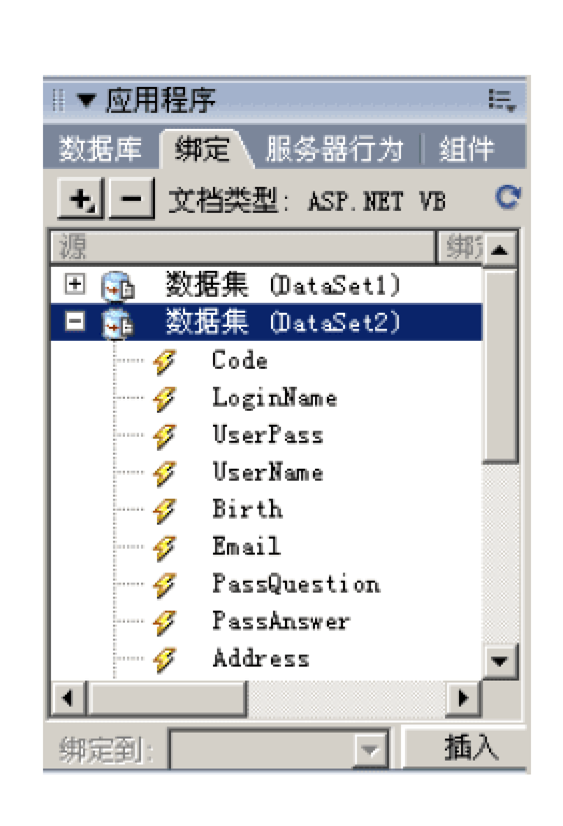


图 10.112　【绑定】面板

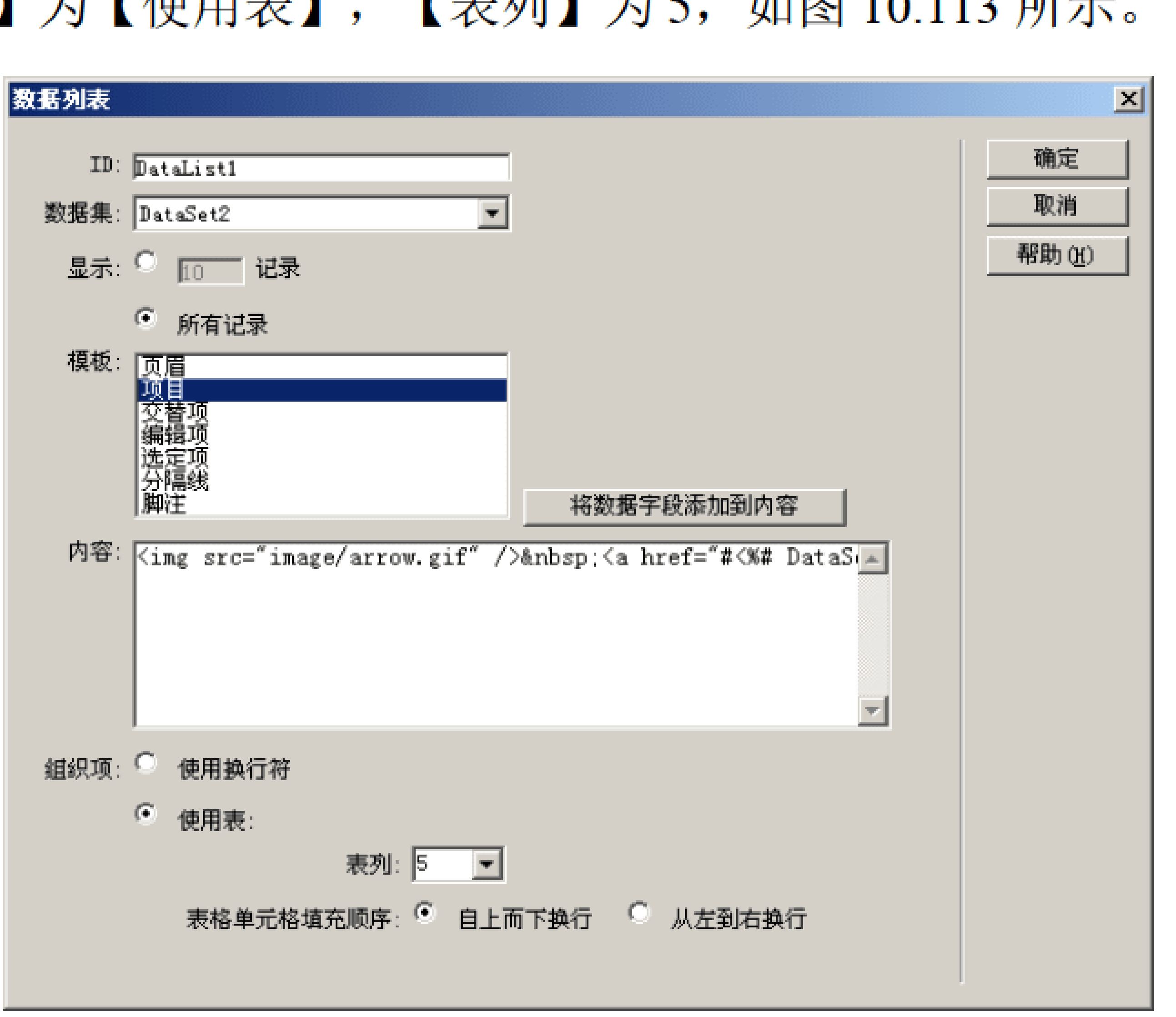


图 10.113　【数据列表】对话框

提示：这里使用了锚点来创建同一个页面内的跳转链接，锚点的名称为同学编码。

(4) 单击【确定】按钮，完成数据列表的创建。在【设计】视图中选择该数据列表并右击，从弹出的快捷菜单中选择【编辑标签】命令，在弹出的【标签编辑器】对话框中，选择【布局】分类，设置其宽度为 90%，单元格边距为 3，如图 10.114 所示。

(5) 单击【确定】按钮，完成数据列表的设置，同时也完成了同学列表的数据绑定。

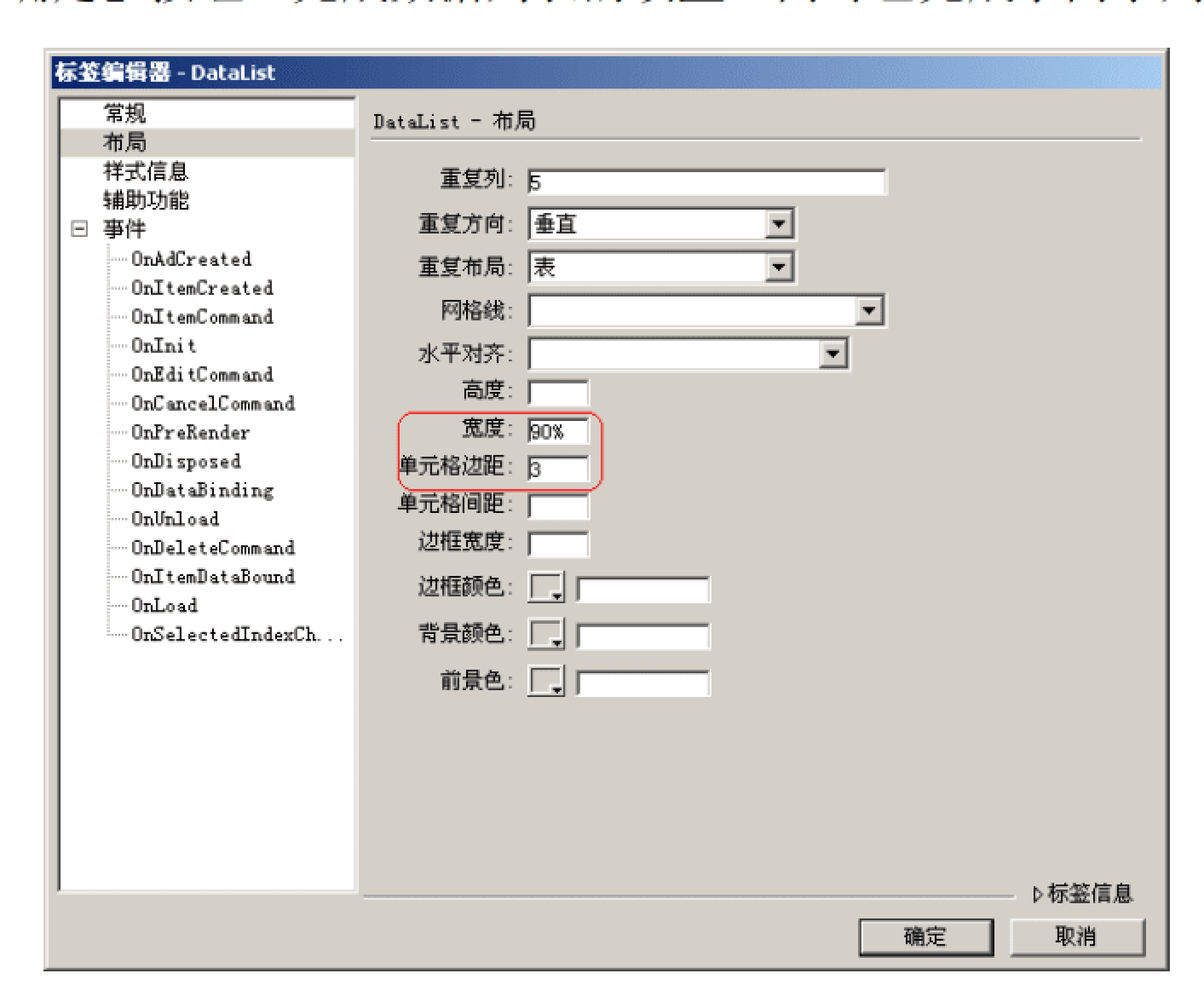


图 10.114　设置数据列表的布局

接下来，将实现所有同学的通讯录信息的浏览。由于数据集 DataSet2 中已经包含了所有同学的基本信息，因此这里的数据绑定将与同学列表一起共用数据集 DataSet2。

3. 动态数据的显示

接下来，实现单条数据的显示。

(1) 在【设计】视图中，将光标置于表格 Table3 之后。在【常用-插入】工具栏中，单击【命名锚记】按钮，此时将弹出【命名锚记】对话框，如图 10.115 所示。

图 10.115　【命名锚记】对话框

(2) 这里暂不设置锚记名称，单击【确定】按钮，创建一个新的锚点。切换至【代码】视图，找到锚点的定义代码，设置其属性 name 的值为“<%# DataSet2.FieldValue("Code", Container) %>”，如图 10.116 所示。

图 10.116　设置锚点的 name 属性

提示：由于锚点的名称只能包含字母和数字，而不能包含特殊符号。因此，在图 10.115 中不能直接输入以上设置值，而只能在代码中进行修改。

(3) 在锚点的起始标记与结束标记之间，插入一个 5 行 4 列的表格，表格属性设置如图 10.117 所示。

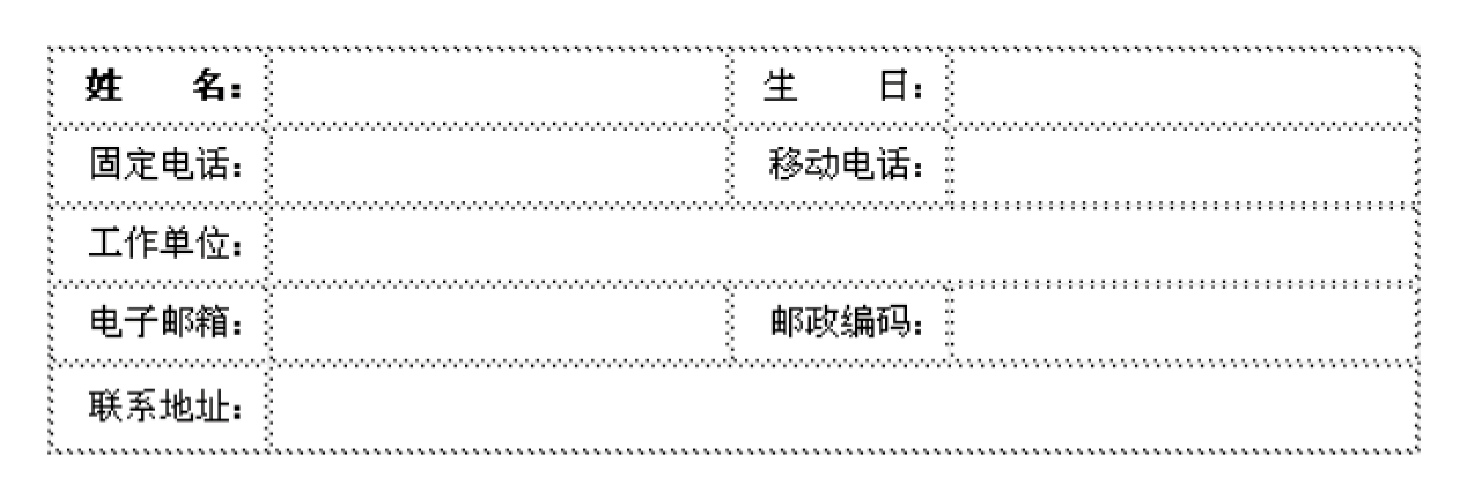


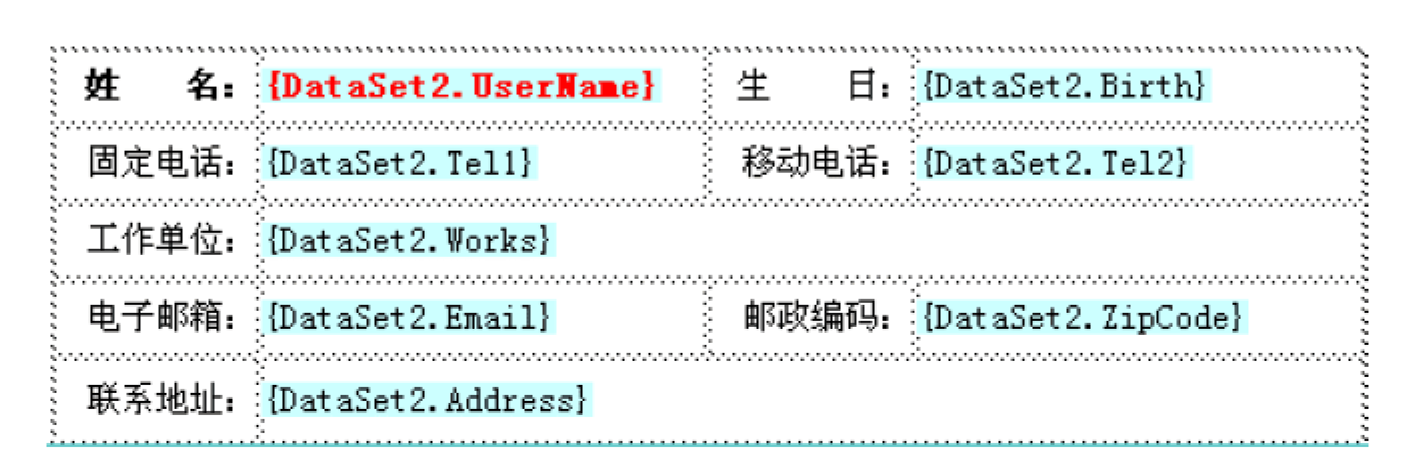
图 10.117　表格属性

(4) 在表格中输入文本，并合并相应的单元格，使之如图 10.118 所示。

姓　名：		生　日：	
固定电话：		移动电话：	
工作单位：			
电子邮箱：		邮政编码：	
联系地址：			

图 10.118　表格设置

(5) 该表格用于显示单个同学的通讯录信息。在【应用程序】面板组中，切换至【绑定】面板，展开数据集 DataSet2，选择并拖动相应的字段至表格中相应的位置，如图 10.119 所示。



姓　名：	**{DataSet2.UserName}**	生　日：	{DataSet2.Birth}
固定电话：	{DataSet2.Tel1}	移动电话：	{DataSet2.Tel2}
工作单位：	{DataSet2.Works}		
电子邮箱：	{DataSet2.Email}	邮政编码：	{DataSet2.ZipCode}
联系地址：	{DataSet2.Address}		

图 10.119　绑定字段

其中，姓名所对应的动态文本(DataSet2.UserName)的字体颜色设置为红色，并加粗显示。在数据表 UserSheet 中，字段 Birth(出生日期)定义为日期/时间类型，其显示将包括年月日和时间。但这里，只需要显示其年月日即可。为此，需要设置该动态文本的格式。

(6) 在【服务器行为】面板中，双击动态文本 DataSet2.Birth，在弹出的【动态文本】对话框中，设置其格式为【日期/时间-2001 年 1 月 17 日】，如图 10.120 所示。

以上操作实现了单个同学信息的显示，下面的操作将实现多个同学信息的显示。

(7) 在【设计】视图中，单击锚点标记 (此操作将同时选择锚点标记内的整个表格)。在【服务器行为】面板中，单击+按钮，从弹出的菜单中选择【重复区域】命令。在弹出的【重复区域】对话框中选择数据集为 DataSet2，并设置显示所有记录，如图 10.121 所示。

图 10.120　设置动态文本格式　　　　图 10.121　重复区域

(8) 单击【确定】按钮，完成重复区域的创建。

至此，页面功能全部设计完成，页面预览效果如图 10.122 所示。

图 10.122　页面预览效果

10.7　班 级 管 理

10.7.1　班级管理

在同学录的主页上，单击【班级管理】图片链接，即可进入当前班级的管理功能页面。在班级管理中，可分为以下 4 种情况。

- 当前班级尚无管理员，此时用户可申请担任本班级的管理员。
- 当目前某位同学已申请担任本班级的管理员，但尚未得到班级多数同学的同意，此时将显示相关信息，并提示当前的同学同意情况。
- 当目前某位同学已成功担任本班级的管理员，但非当前用户，此时将显示相关信息。
- 当目前某位同学已成功担任本班级的管理员，且管理员就是当前用户，此时将显示同学列表，并允许管理员对指定同学执行踢出班级操作。

对于以上 4 种情况，在页面中将通过 4 个 Panel 控件来包含各自所要显示的内容，并通过代码来控制当前显示哪种情况，即显示哪个 Panel 控件。

(1) 新建一个 ASP.NET VB 动态页，将其命名为 Bjgl.aspx。选择【修改】|【模板】|【套用模板到页】命令，在弹出的【选择模板】对话框中，选择【页面模板】。单击【确定】按钮，将模板应用到本页面。

(2) 在表单 Form 中插入一个 1 行 1 列的表格，用于显示当前页面所在的班级，如图 10.123 所示。

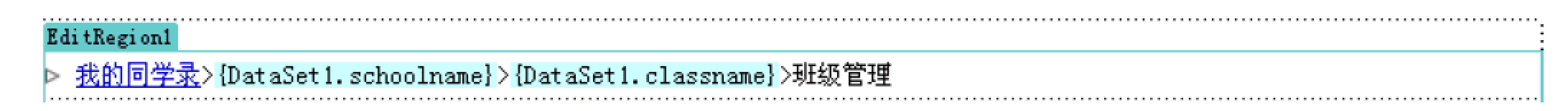


图 10.123　表格设计

这里需要创建一个数据集 DataSet1 来实现学校名称及班级名称的动态显示。该数据集与班级通讯录页面中所定义的数据集 DataSet1 完全一样，因此对其创建过程这里不再赘述。

(3) 创建页面的第一个 Panel 控件。该控件用于显示当前班级尚无管理员情况下的相关信息。单击【ASP.NET-插入】工具栏中的按钮，在弹出的【标签选择器】对话框中，选择【ASP.NET 标签】|【Web 服务器】分类，并在右边的列表框中选择 asp:Panel。单击【插入】按钮，此时将弹出【标签编辑器】对话框。在【常规】选项中，设置其 ID 属性为 Panel1，如图 10.124 所示。

图 10.124　【标签编辑器】对话框

单击【确定】按钮，完成 Panel 控件的插入。

(4) 在 Panel 控件 Panel1 中，插入一个 3 行 3 列的表格，其属性设置如图 10.125 所示。

分别合并表格第 1 行和第 3 行的 3 个单元格，并分别设置其背景图片为 Image 目录下的 lybg_1.gif 和 lybg_2.gif。然后，分别设置表格第 2 行的第 1 个单元格和第 3 个单元格的背景图片为 Image 目录下的 lybg_3.gif 和 lybg_4.gif。此操作用于设置显示信息的边框。

图 10.125　表格属性

(5) 在表格第 2 行的第 2 个单元格中再次插入一个 4 行 1 列的表格，该表格用于控制相关信息的显示，其表格样式如图 10.126 所示。

图 10.126　表格设计

(6) 在表格所显示的信息中，插入了一个动态文本 DataSet1.ClassName，用于显示当前的班级名称。此外，在表格中还添加了一个按钮控件和一个标签控件。其中，标签控件用于显示当申请操作成功时的提示信息。单击【我要申请】按钮，可执行申请班级管理员的相关操作，该按钮的 ID 设置为 Sure，单击事件为 Sure_Click。

(7) 切换至【代码】视图，添加【我要申请】按钮的单击事件，其代码如下：

【示例代码】

```
Sub Sure_Click(ByVal Sender As Object, ByVal E As EventArgs)
  Dim Cnn As OleDbConnection
  Dim Cmd As OleDbCommand
  Dim StrCnn As String
  Dim Sql As String
  '获取数据库连接字符串
  StrCnn =
System.Configuration.ConfigurationSettings.AppSettings("MM_CONNECTION_ST
RING_Cnn")
  Cnn = New OleDbConnection(StrCnn)
  Cnn.Open()  '创建数据库连接
  '更新数据表 ClassInfo，将申请标志 shtag 设置为 1，同时将申请管理员 adminuser 设置为
当前用户
  Sql="update classinfo set adminuser=" & trim(request.Cookies("id").value)
& ",shtag='1' where id=" & trim(request.QueryString("id"))
  Cmd=New OleDbCommand(sql,Cnn)
  Cmd.ExecuteNonquery  '执行更新操作
```

```
    Cnn.close()
    '显示申请成功后的提示信息
    LblInfo.Text="您的申请已经提交，请静候其他同学的批复！"
    '将【我要申请】按钮置为不可用状态，以免用户再次单击提交
    Sure.Enabled=False
End sub
```

至此，Panel 控件 Panel1 的功能设计完成。

下面，添加一个新的 Panel 控件，用于显示当某位同学已申请担任本班级的管理员，但尚未得到班级多数同学的同意批复情况下的相关信息，即此时班级管理员处于待定状态。

(8) 在【代码】视图中，选择并复制整个 Panel1 控件的代码，然后在其后进行粘贴。切换至【设计】视图，对复制生成的 Panel 控件进行修改，设置其 ID 为 Panel2。然后，删除用于显示信息的嵌套表格的第 4 行，并对该表格的内容进行调整，调整后的表格如图 10.127 所示。

图 10.127　表格设计

这里，除了对显示文本进行修改外，还添加了 4 个标签控件和一个【关闭】按钮控件。标签控件分别用于显示当前班级中申请担任班级管理员的同学姓名，以及在同学批复情况中已经同意、未同意和尚未批复的同学人数，标签控件的值将在页面的 Page_Load 事件中进行设置。

(9) 单击【关闭】按钮，可关闭当前页面，该按钮的 ID 设置为 Close，单击事件为 Close_Click。切换至【代码】视图，添加【关闭】按钮的单击事件，其代码如下：

【示例代码】

```
Sub Close_Click(ByVal Sender As Object, ByVal E As EventArgs)
  ClientScript.RegisterStartupScript(Me.GetType(), "",CloseWindow())
End Sub
```

在该事件中，仅包含一个语句，即调用 ClientScript.RegisterStartupScript()方法来调用自定义函数 CloseWindow()所返回的 Script 脚本，该脚本的功能是关闭当前页面。

CloseWindow()函数的定义如下：

【示例代码】

```
Function CloseWindow() as String
  dim Str2 as String
  str2="<script language='javascript'>window.close();<"
  str2+="/"
  str2+="script>"
  return trim(str2)
End Function
```

至此，Panel 控件 Panel2 的功能设计完成。

接下来，再通过复制 Panel2 控件来生成一个新的 Panel 控件，并修改其 ID 为 Panel3。该控件将对应于第 3 种情况，即当目前某位同学已成功担任本班级的管理员，但非当前用户。

(10) 在 Panel3 控件中，修改所显示的文本信息，并将【关闭】按钮控件的 ID 设置为 Close1，其事件名不变。

提示： 在同一页面中，不能存在两个 ID 名相同的 ASP.NET 控件。但不同的控件，可调用同一事件。

修改后的表格如图 10.128 所示。

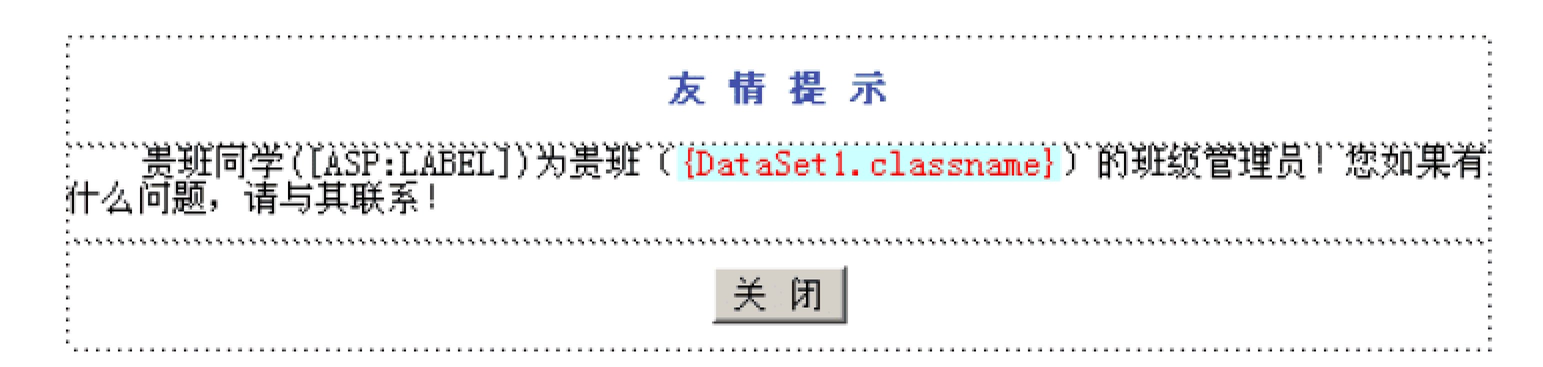


图 10.128　表格设计

在此表格中，【关闭】按钮将执行 Panel2 控件中的【关闭】按钮所执行的事件，即 Close_Click 事件。

对于第 4 个 Panel 控件，由于其界面显示与其他 Panel 控件不同，因此不能通过复制来快速生成。在第 4 种情况下，即当前用户就是本班级的管理员，此时应提供对当前班级所有同学的管理。管理同学的主要操作为踢出班级操作，该操作与主页面中的“退出此班级”操作类似，惟一的区别是后者是用户主动退出班级，而前者是管理员强制其退出。

(11) 单击【ASP.NET-插入】工具栏中的按钮，在弹出的【标签选择器】对话框中，选择【ASP.NET 标签】|【Web 服务器】分类，并在右边的列表框中选择 asp:Panel。单击【插入】按钮，添加一个新的 Panel 控件，并设置其 ID 为 Panel4。

为了绑定同学列表的显示，这里需要创建一个新的数据集。

(12) 打开【服务器行为】面板，选择数据集 DataSet1，将其复制并以粘贴生成新的数据集 DataSet2。双击数据集 DataSet2，在弹出的【数据集】对话框中，将 SQL 语句修改如下：

```
SELECT code,username,sex,tell,address,nums,currlogin FROM usersheet WHERE
code in (select usercode from userclass where classcode=?) ORDER BY code
```

如图 10.129 所示。

单击【确定】按钮，完成数据集 DataSet2 的设置。

(13) 将光标置于控件 Panel4 中，在【服务器行为】面板中，单击按钮，在弹出的菜单中选择【数据网格】命令，在弹出的【数据网格】对话框中，设置 ID 为 DataGrid1，数据集选择 DataSet2，并设置为一次显示 10 条记录，【导航】选项设置为【编号链接到每一页】，如图 10.130 所示。

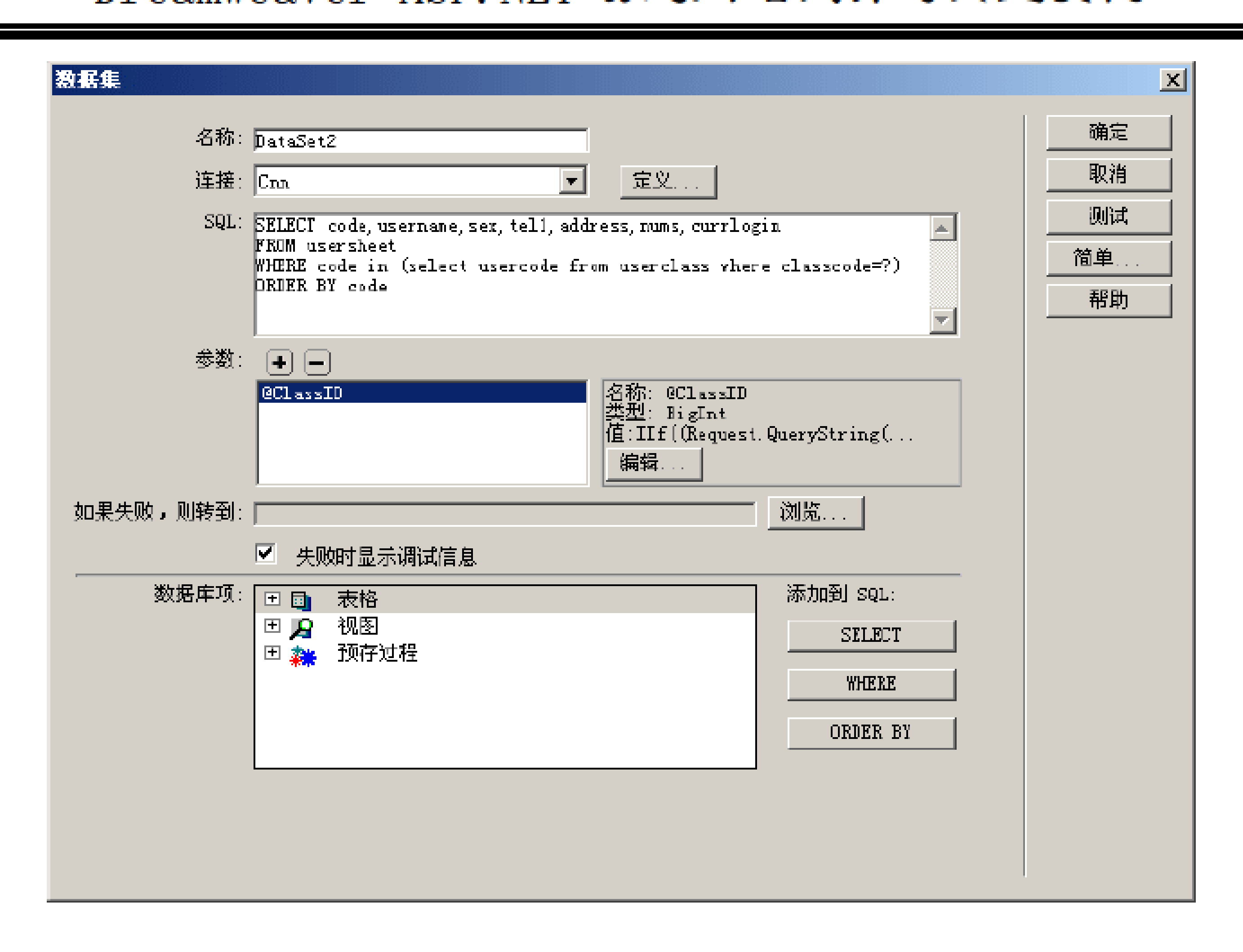


图 10.129　编辑数据集 DataSet2

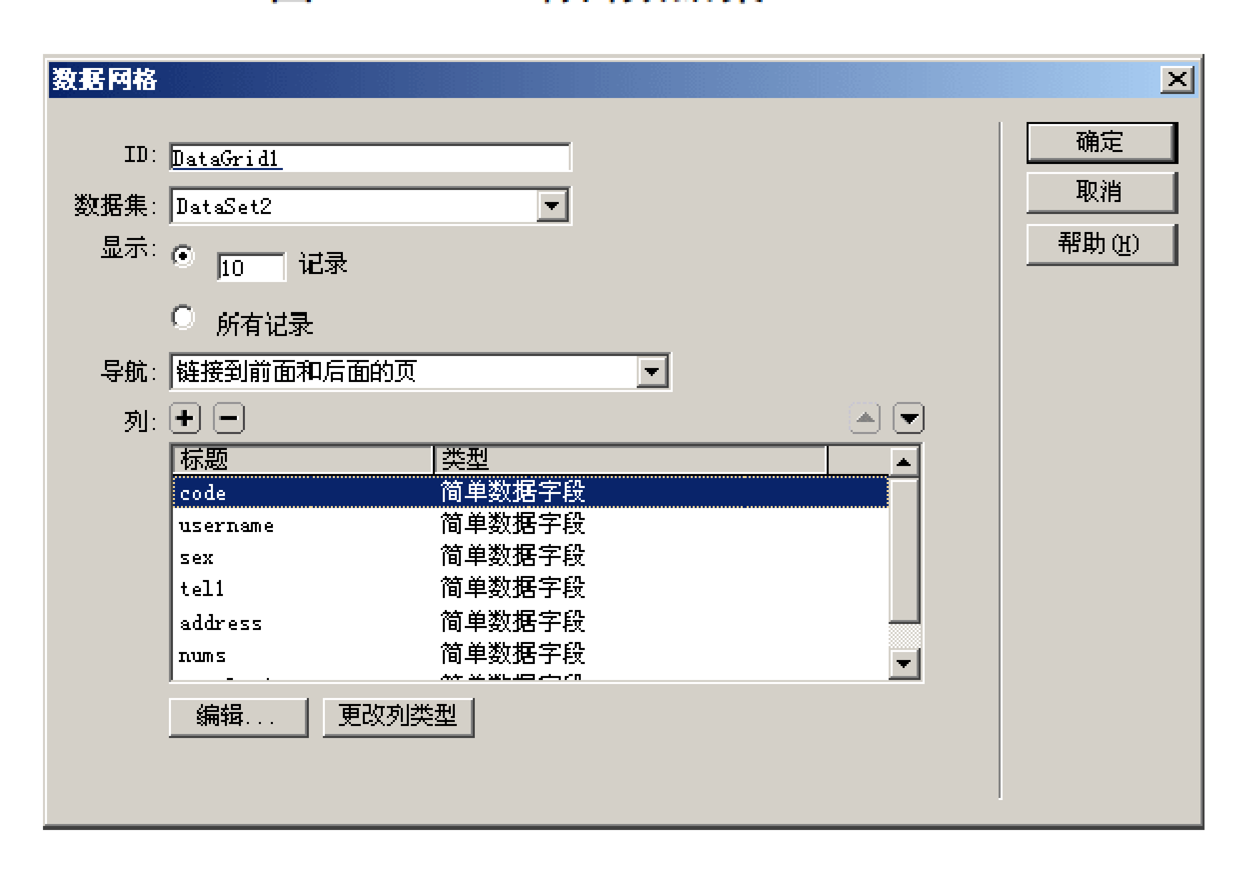


图 10.130　【数据网格】对话框

(14) 单击【添加列】按钮，从弹出的菜单中选择【'删除'按钮】命令，在弹出的【删除按钮列】对话框中，设置显示标题为【踢出班级】，【按钮类型】选择【链接按钮】，【删除自】选择 UserSheet，【主键】选择 Code，【提交为】选择 Integer，如图 10.131 所示。

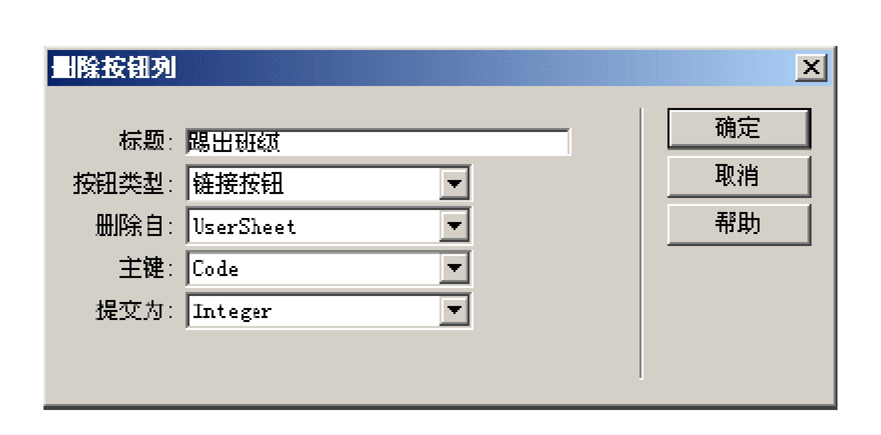


图 10.131　【删除按钮列】对话框

单击【确定】按钮，完成删除列的添加。注意，这里所添加的删除列并非真正删除用户的信息，而是删除用户与班级的关联。为此，后面将会对删除列的默认事件进行修改。

(15) 在【数据网格】对话框的【列】列表框中，选择 ID 列，单击【编辑】按钮，在弹出的【简单数据字段列】对话框中设置该列的显示标题为“编码”，如图 10.132 所示。

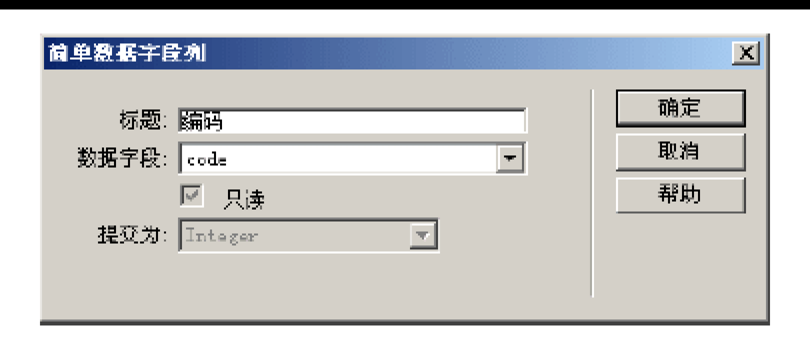


图 10.132　设置 ID 列的属性

(16) 重复以上操作，依次设置其余各列的标题。设置后的【数据网格】对话框如图 10.133 所示。

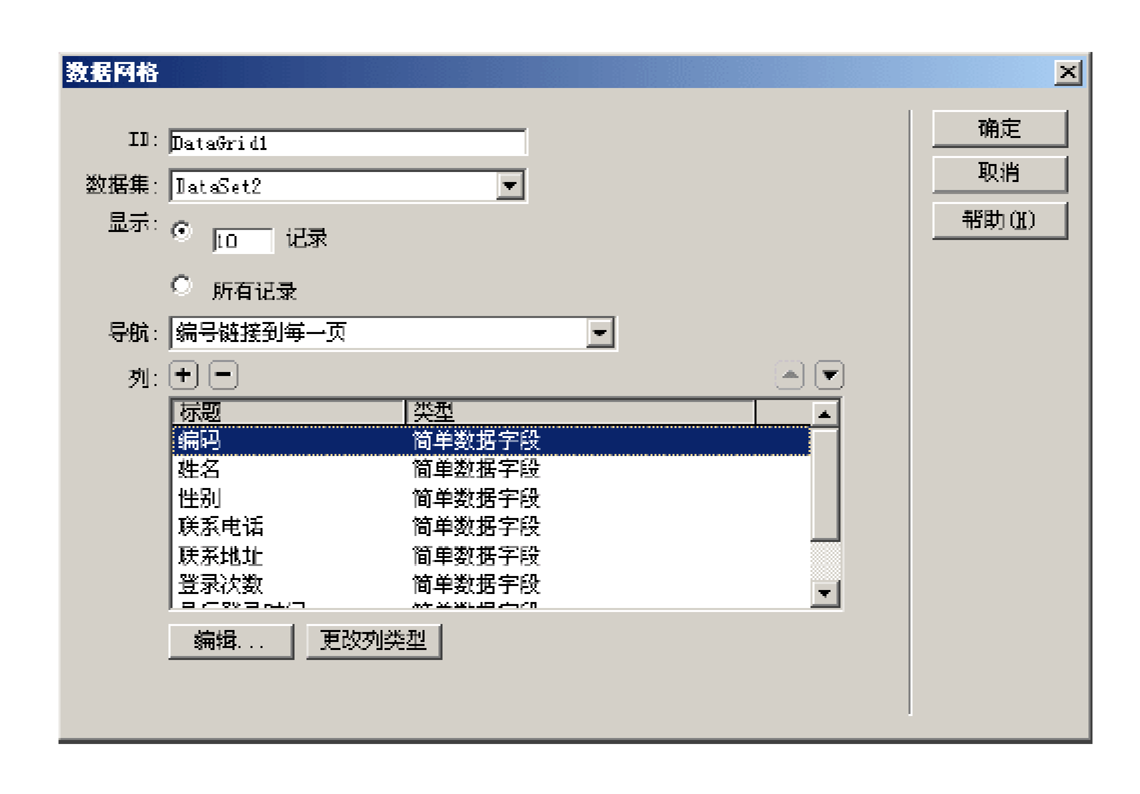


图 10.133　数据网格中列的设置

单击【确定】按钮，完成数据网格的初步创建。

(17) 此时，还需要对数据网格的显示作进一步调整。在【设计】视图中，选择所创建的数据网格并右击，从弹出的快捷菜单中选择【编辑标签】命令，在弹出的【标签编辑器】对话框中，选择【布局】分类，设置其宽度为 90%，单元格边距为 3，单元格间距为 0，边框宽度为 1。然后，选择【事件】分类中的 OnItemCommand，将其默认的事件 DataSet1.OnDataGridDelete 改为自定义的事件 DataGrid1_Mod，如图 10.134 所示。

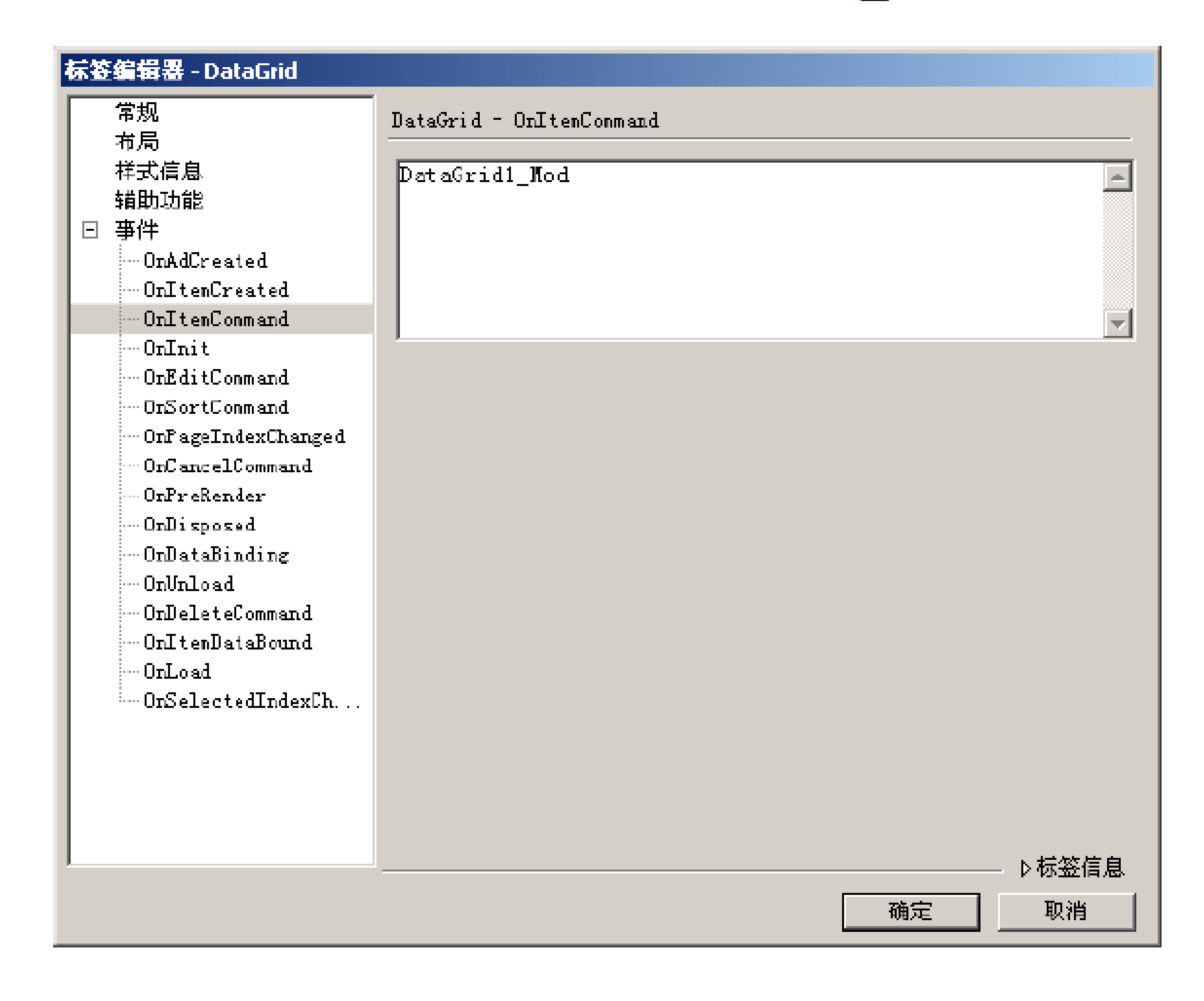


图 10.134　设置 OnItemCommand 事件

单击【确定】按钮，完成数据网格的事件设置。

(18) 切换至【代码】视图，添加 DataGrid1_Mod 事件代码，如下：

【示例代码】

```
Sub DataGrid1_Mod(ByVal Sender As Object, ByVal E As
DataGridCommandEventArgs)
  Dim codestr As String = E.Item.Cells(0).Text  '获取所要踢出班级的同学 ID
  Dim Cnn As OleDbConnection
  Dim Cmd As OleDbCommand
  Dim StrCnn As String
  Dim Sql As String
  If CType(e.CommandSource, LinkButton).CommandName = "Delete" Then
    '获取数据库连接字符串
    StrCnn =
System.Configuration.ConfigurationSettings.AppSettings("MM_CONNECTION_ST
RING_Cnn")
    '创建数据库连接
    Cnn = New OleDbConnection(StrCnn)
    Cnn.Open()
    '删除对指定用户在本班级中上传的相片所发表的评论信息
    Sql = "delete from photo_pl where photoid in (select id from photoinfo
where upuser=" & trim(codestr) & " and classcode=" &
trim(request.QueryString("id")) & ")"
    Cmd = New OleDbCommand(Sql, Cnn)
    Cmd.ExecuteNonquery
    '删除指定用户在本班级中所上传的相片信息
    Sql = "delete from photoinfo where upuser=" & trim(codestr) & " and
classcode=" & trim(request.QueryString("id"))
    Cmd = New OleDbCommand(Sql, Cnn)
    Cmd.ExecuteNonquery
    '删除指定用户与本班级的关联
    Sql = "delete from userclass where usercode=" & trim(codestr) & " and
classcode=" & trim(request.QueryString("id"))
    Cmd = New OleDbCommand(Sql, Cnn)
    Cmd.ExecuteNonquery
    '如果指定用户为本班级的管理员，则将班级的管理员信息清空
    Sql = "update classinfo set shtag='0' where id=" &
trim(request.QueryString("id")) & " and adminuser=" & trim(codestr)
    Cmd = New OleDbCommand(Sql, Cnn)
    Cmd.ExecuteNonquery
    Cnn.Close()  '关闭数据库连接
    '将页面跳转至本页面，刷新数据的显示
    response.Redirect("Bjgl.aspx?id=" & request.QueryString("id"))
  End If
End Sub
```

在 DataGrid1_Mod 事件中，除了需要删除指定用户与当前班级的关联外，还需要删除该用户在当前班级的其他信息，包括所上传的相片以及针对相片的相关评论。

最后，来看页面加载时所执行的 Page_Load 事件。该事件在本页面中至关重要，它控制了页面的不同显示。在 Page_Load 事件中，首先需要判断当前班级的管理员情况。如果

班级已经存在管理员，则还需判断当前用户是否就是班级的管理员，并针对不同的情况，来显示不同的界面。

以下，是 Page_Load 事件的相关代码：

【示例代码】

```
Sub Page_Load(ByVal Sender As Object, ByVal E As EventArgs)
  Dim Cnn As OleDbConnection
  Dim Cmd As OleDbCommand
  Dim StrCnn As String
  Dim DataR AS OleDbDataReader
  Dim Sql As String
  '定义变量 Num_Agree，用于记录同学批复中的同意人数
  Dim Num_Agree as integer
  '定义变量 Num_NotAgree，用于记录同学批复中的未同意人数
  Dim Num_NotAgree As integer
  '定义变量 Num_Other，用于记录同学批复中的尚未批复人数
  Dim Num_Other As Integer
  Dim SQUser As String
  Dim i As Integer
  If Not IsPostBack() Then
    '判断 Cookie 变量 ID 及页面变量 ID 是否为空，前者表示当前用户，后者表示当前班级
    If IsDBNull(Request.Cookies("id").Value) Or
Trim(Request.Cookies("id").Value) = "" Or Request.QueryString("id")="" Then
      'Cookie 变量 ID 或页面变量 ID 为空，将页面跳转至登录页面
      Response.Redirect("default.aspx")
    Else
      '获取数据库连接字符串
      StrCnn =
System.Configuration.ConfigurationSettings.AppSettings("MM_CONNECTION_ST
RING_Cnn")
      Cnn = New OleDbConnection(StrCnn)
      Cnn.Open()
      '查询当前班级是否存在管理员信息，包括待批复的和已经成功担任的
      Sql="SELECT code,username,shtag FROM usersheet,classinfo WHERE
adminuser=usersheet.code and shtag<>'0' and classinfo.id=" &
trim(request.QueryString("id"))
      Cmd=New OleDbCommand(Sql,Cnn)
      DataR=Cmd.ExecuteReader
      If DataR.Read() Then
        '存在班级管理员信息，判断其状态是待批复还是成功担任
        If DataR("ShTag")="1" Then
          '班级管理员的状态为待批复，仅显示 Panel2 控件
          Panel1.Visible=False
          Panel2.Visible=True
          Panel3.Visible=False
          Panel4.Visible=False
          LblTx.Text=DataR("username")  '获取并显示申请班级管理员的同学姓名
          SQUser=DataR("code")  '获取申请班级管理员的同学 ID
          DataR.Close()
          '查询其他同学的批复情况
```

```
          Sql="select hf_admin,count(*) as nums from userclass where 
usercode<>" & trim(SQUser) & " and classcode=" & 
trim(request.QueryString("id")) & " group by hf_admin "
          Cmd=New OleDbCommand(Sql,Cnn)
          DataR=Cmd.ExecuteReader
          While DataR.Read()
            If DataR("hf_admin")="1" Then
              '获取不同意的同学人数
              Num_NotAgree=DataR("Nums")
            ElseIf DataR("hf_admin")="2" Then
              '获取同意的同学人数
              Num_Agree=DataR("Nums")
            Else
              '获取尚未批复的同学人数
              Num_Other=DataR("Nums")
            End If
          End While
          LblAgree.Text=Num_Agree.ToString()
          LblNotAgree.Text=Num_NotAgree.ToString()
          LblOther.Text=Num_Other.ToString()
        ElseIF DataR("code")<>request.Cookies("id").value then
          '班级管理员的状态为成功担任，且管理员非当前用户
          '仅显示 Panel3 控件
          Panel1.Visible=False
          Panel2.Visible=False
          Panel3.Visible=True
          Panel4.Visible=False
          LblTx1.Text=DataR("username")
        Else
          '班级管理员的状态为成功担任，且当前用户就是班级管理员
          '仅显示 Panel4 控件
          Panel1.Visible=False
          Panel2.Visible=False
          Panel3.Visible=False
          Panel4.Visible=True
        End If
      Else
        '尚无班级管理员信息，显示 Panel1 控件
        Panel1.Visible=True
        Panel2.Visible=False
        Panel3.Visible=False
        Panel4.Visible=False
      End If
      Cnn.Close()  '关闭数据库连接
    End If
  End If
End Sub
```

至此，班级管理页面的功能设计全部完成。根据不同的情况给出不同的显示。在当前班级尚无管理员时，页面预览效果如图 10.135 所示。

图 10.135　页面预览效果

此时，单击【我要申请】按钮，可申请担任当前班级的班级管理员。在当前班级已有管理员但尚处于待定状态时，页面预览效果如图 10.136 所示。

图 10.136　页面预览效果

单击【关闭】按钮，可关闭当前页面。当当前班级已存在管理员信息且已成功担任时，如果当前用户非班级管理员，则页面预览效果如图 10.137 所示。

如果当前用户为当前班级的班级管理员，则页面预览效果如图 10.138 所示。

此时，可对指定的同学执行踢出班级操作，即将该用户从本班级中删除。

图 10.137　页面预览效果

图 10.138　页面预览效果

10.7.2　批复申请

当某个同学申请担任指定班级的管理员时，只有经过该班大多数同学的同意之后，才可正式成为该班的班级管理员；而如果多数同学批复不同意，则该用户的申请将被取消。

对于管理员申请的批复操作是在一个弹出页面中执行的，页面文件名为 Pop_Info.aspx。当用户进入同学录的主页(此时，将加载显示其最先加入的班级信息)或选择进入指定的班级时，在页面的 Page_Load 事件中将会判断该班级是否存在待批复的管理员信息；如果存在，则弹出批复申请页面。为此，需要修改系统主页(Main.aspx)中的 Page_Load 事件。修改后的事件代码如下：

【示例代码】

```
Sub Page_Load(ByVal Sender As Object, ByVal E As EventArgs)
  Dim Cnn As OleDbConnection
```

```
  Dim Cmd As OleDbCommand
  Dim StrCnn As String
  Dim DataR AS OleDbDataReader
  Dim Sql As String
  Dim Pop_Str As String  '定义变量 Pop_Str，用于存储弹出页面的 Script 脚本
  If Not IsPostBack() Then
    If IsDBNull(Request.Cookies("id").Value) Or
Trim(Request.Cookies("id").Value) = "" Then
      'Cookie 变量 ID 为空，将页跳转至登录页面
      Response.Redirect("default.aspx")
    Else
      if Request.QueryString("ClassID")="" then
        '页面参数 ClassID 为空，获取数据集 DataSet2 的第一条记录的字段 ID 的值作为默认
的班级编码
        If DataSet2.DefaultView.Table.Rows.Count>0 Then
          Response.Redirect("main.aspx?ClassID=" &
DataSet2.DefaultView.Table.Rows(0)("ID"))
        End if
      Else
        '获取数据库连接字符串
        StrCnn =
System.Configuration.ConfigurationSettings.AppSettings("MM_CONNECTION_ST
RING_Cnn")
        Cnn = New OleDbConnection(StrCnn)
        Cnn.Open()
        '查询当前班级是否有需要批复的管理员申请信息
        Sql="Select ClassInfo.* from ClassInfo,UserClass where
ClassInfo.Id=ClassCode and shtag='1' and ClassInfo.Id=" &
trim(request.Querystring("ClassID")) & " and UserCode=" &
trim(Request.Cookies("id").Value) & " and HF_Admin='0' and AdminUser<>" &
trim(Request.Cookies("id").value)
        Cmd=New OleDbCommand(sql,Cnn)
        DataR=Cmd.ExecuteReader()
        If DataR.Read() Then
          '存在待批复的管理员申请信息
          '设置用于弹出页面的 Script 脚本
          Pop_Str="<script language=javascript>"
          Pop_Str=Pop_Str & "window.open('Pop_Info.aspx?ClassId=" &
trim(Datar("id")) & "','info',"
          Pop_Str=Pop_Str &
    "'width=420px,height=300px,toolbar=no,menubar=no,left=0,top=0,scroll
bars=yes,resizable=yes,location=no,status=no');</"
          Pop_Str=Pop_Str & "script>"
          '执行 Script 脚本，弹出批复申请页面
          ClientScript.RegisterStartupScript(Me.GetType(), "",Pop_Str)
        End If
        Cnn.Close()  '关闭数据库连接
      End if
    End If
  End If
End Sub
```

下面，来看批复申请页面的具体实现。

(1) 新建一个 ASP.NET VB 类型的页面，将其命名为 Pop_Info.aspx。由于该页面为弹出窗口页面，因此不需要加载页面模板。

(2) 在页面中添加一个表单 Form，并设置其 Runat 属性为 Server。然后，在表单 Form 中插入一个 3 行 1 列的表格，并在表格中添加相应的文本和控件，如图 10.139 所示。

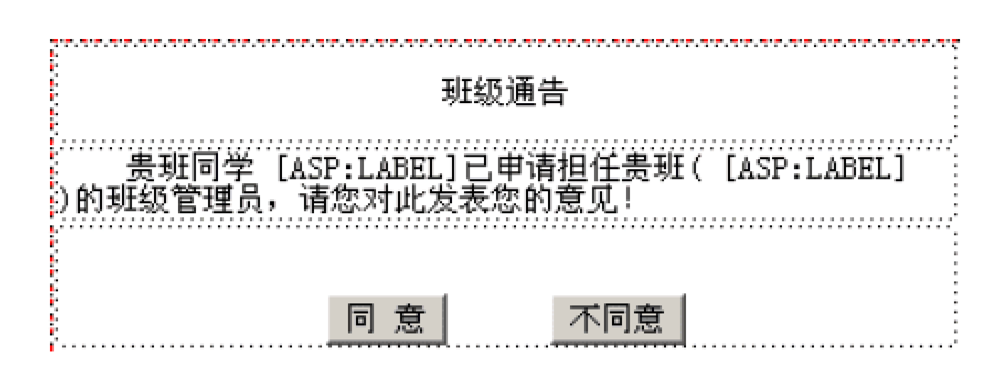


图 10.139　表格设计

(3) 设置表格的宽度为 370px，并设置文本“班级通告”的字体大小为 15 像素，加粗显示。

在表格中，添加了两个标签控件和两个按钮控件。其中，标签控件分别用于显示申请担任班级管理员的同学名称和当前的班级名称，其数据的加载将在页面的 Page_Load 事件中执行。单击【同意】按钮，表示同意该同学对班级管理员的申请，该按钮控件的 ID 为 Agree，单击事件设置为 Agree_Click；单击【不同意】按钮，表示不同意该同学对班级管理员的申请，该按钮控件的 ID 为 NotAgree，单击事件设置为 NotAgree_Click。

下面，来看页面加载时所执行的 Page_Load 事件，其代码如下：

【示例代码】

```
Sub Page_Load(ByVal Sender As Object, ByVal E As EventArgs)
  Dim Cnn As OleDbConnection
  Dim Cmd As OleDbCommand
  Dim StrCnn As String
  Dim DataR AS OleDbDataReader
  Dim Sql As String
  If Not IsPostBack() Then
    '获取数据库连接字符串
    StrCnn =
System.Configuration.ConfigurationSettings.AppSettings("MM_CONNECTION_ST
RING_Cnn")
    Cnn = New OleDbConnection(StrCnn)
    Cnn.Open()  '打开数据库连接
    '查询当前的班级信息及待批复的管理员信息
    Sql="Select ClassInfo.*,username from ClassInfo,UserSheet,UserClass
where code=adminuser and ClassInfo.Id=ClassCode and shtag='1' and
ClassInfo.Id=" & trim(request.Querystring("ClassID")) & " and UserCode=" &
trim(Request.Cookies("id").Value) & " and HF_Admin='0' and AdminUser<>" &
trim(Request.Cookies("id").value)
    Cmd=New OleDbCommand(sql,Cnn)
    DataR=Cmd.ExecuteReader()
    If DataR.Read() Then
      '获取申请班级管理员的同学姓名
      Label1.Text=DataR("UserName")
      '获取当前班级名称
```

```
      Label2.Text=DataR("ClassName")
      '获取申请班级管理员的同学 ID，并将其赋予全局共享变量 SQStr
      SQStr=DataR("AdminUser")
    End If
    Cnn.Close()  '关闭数据库连接
  End If
End Sub
```

该页的 Page_Load 事件主要用来查询当前的班级信息及申请班级管理员的同学姓名，从而绑定页面上相应标签控件的数据显示。

对于两个按钮控件的单击事件，其代码如下：

【示例代码】

```
'【同意】按钮的单击事件
Sub Agree_Click(ByVal Sender As Object, ByVal E As EventArgs)
  PF_SQ(2)
End Sub
'【不同意】按钮的单击事件
Sub NotAgree_Click(ByVal Sender As Object, ByVal E As EventArgs)
  PF_SQ(1)
End Sub
```

在以上事件代码中，仅仅是调用了同一个自定义过程 PF_SQ()来执行相应的数据操作。对于不同的按钮事件，所传递的参数不同。

下面，来看自定义过程 PF_SQ()中的代码。

【示例代码】

```
Sub PF_SQ(Byval Tag As Integer)
  '变量 Tag 表示当前所执行的操作类别，其值为 1 表示执行不同意操作，为 2 表示执行同意操作
  Dim Cnn As OleDbConnection
  Dim Cmd As OleDbCommand
  Dim DataR AS OleDbDataReader
  Dim StrCnn As String
  Dim Sql As String
  Dim Num_Agree As Integer  '定义变量 Num_Agree，用于存储同意批复的人数
  Dim Num_NotAgree As Integer '定义变量 Num_NotAgree，用于存储不同意批复的人数
  Dim Num_Other As Integer  '定义变量 Num_Other，用于存储尚未批复的人数
  Dim Num_Total As Integer  '定义变量 Num_Total，用于存储除申请人之外的班级总人数
  '获取数据库连接字符串
  StrCnn =
System.Configuration.ConfigurationSettings.AppSettings("MM_CONNECTION_ST
RING_Cnn")
  Cnn = New OleDbConnection(StrCnn)
  Cnn.Open()  '打开数据库连接
  '统计当前对班级管理员申请的批复情况
  Sql="select hf_admin,count(*) as nums from userclass where usercode<>" &
trim(SQStr) & " and classcode=" & trim(request.QueryString("classid")) & "
group by hf_admin"
  Cmd=New OleDbCommand(Sql,Cnn)
  DataR=Cmd.ExecuteReader
  While DataR.Read()
```

```
        If DataR("hf_admin")="1" Then
          Num_NotAgree=DataR("Nums")  '获取不同意人数
        ElseIf DataR("hf_admin")="2" Then
          Num_Agree=DataR("Nums")  '获取同意人数
        Else
          Num_Other=DataR("Nums")  '获取尚未批复的人数
        End If
      End While
      DataR.Close()
      '获取可对班级管理员申请进行批复的总人数
      Num_Total=Num_NotAgree+Num_Agree+Num_Other
      If Tag=1 Then
        'Tag 的值为 1，表示执行的是不同意当前对班级管理员的申请
        Num_NotAgree=Num_NotAgree+1  '不同意人数加 1
        if (Num_NotAgree/Num_Total)>0.5 Then
           '当不同意人数超过总人数的一半时，班级管理员的申请失败
           '清空班级信息表 ClassInfo 中管理员申请信息
           Sql="Update ClassInfo set AdminUser=0,shtag='0' where id=" &
    trim(request.QueryString("classid"))
          Cmd=New OleDbCommand(Sql,Cnn)
          Cmd.ExecuteNonquery
          '将以前其他同学对班级管理员申请的批复情况置为初始状态
          Sql="Update UserClass set hf_admin='0' where classcode=" &
    trim(request.QueryString("classid"))
          Cmd=New OleDbCommand(Sql,Cnn)
          Cmd.ExecuteNonquery
        Else
          '不同意人数尚未超过总人数的一半，仅更新当前用户的批复情况
          Sql="update userclass set hf_admin='1' where usercode=" &
    trim(request.Cookies("id").Value) & " and classcode=" &
    trim(request.QueryString("classid"))
          Cmd=New OleDbCommand(Sql,Cnn)
          Cmd.ExecuteNonquery
        End if
      ElseIF Tag=2 Then
        "Tag 的值为 1，表示执行的是同意当前对班级管理员的申请
        Num_Agree=Num_Agree+1  '同意人数加 1
        if (Num_Agree/Num_Total)>0.5 Then
          '当同意人数超过总人数的一半时，班级管理员的申请成功
          '将数据表 ClassInfo 中的管理员申请审核标志置为 2，即表示已申请成功
          Sql="Update ClassInfo set shtag='2' where id=" &
    trim(request.QueryString("classid"))
          Cmd=New OleDbCommand(Sql,Cnn)
          Cmd.ExecuteNonquery
          '将以前其他同学对班级管理员申请的批复情况置为初始状态
          Sql="Update UserClass set hf_admin='0' where classcode=" &
    trim(request.QueryString("classid"))
          Cmd=New OleDbCommand(Sql,Cnn)
          Cmd.ExecuteNonquery
        Else
          '同意人数尚未超过总人数的一半，仅更新当前用户的批复情况
```

```
      Sql="update userclass set hf_admin='2' where usercode=" &
trim(request.Cookies("id").Value) & " and classcode=" &
trim(request.QueryString("classid"))
      Cmd=New OleDbCommand(Sql,Cnn)
      Cmd.ExecuteNonquery
    End if
  End If
  Cnn.close()  '关闭数据库连接
  '执行客户端脚本，关闭当前页面
  ClientScript.RegisterStartupScript(Me.GetType(), "",CloseWindow())
End Sub
```

以上代码执行时，首先获取当前对班级管理员申请的批复情况，包括已同意申请的人数、不同意申请的人数、尚未批复的人数以及总人数。然后，根据参数 Tag 的值来判断当前所要执行的操作。

当 Tag 的值为 1 时，表示用户在批复时单击的是【不同意】按钮，此时将不同意人数加 1。如果不同意人数超过总人数的一半，则表示当前对班级管理员的申请失败，应清空相关的班级管理员申请信息；如果不同意人数尚未超过总人数的一半，则仅更新当前用户对班级管理员申请的批复情况。

当 Tag 的值为 2 时，表示用户在批复时单击的是【同意】按钮，此时将同意人数加 1。如果同意人数超过总人数的一半，则表示当前对班级管理员的申请成功，应将 ClassInfo 数据表中标识管理员申请的标志 ShTag 置为 2；如果同意人数尚未超过总人数的一半，则仅更新当前用户对班级管理员申请的批复情况。

在 ClientScript.RegisterStartupScript()方法中所调用的用于执行关闭当前页面操作的自定义函数 CloseWindow()在上一小节中已经进行了说明，这里不再介绍。

至此，已实现了批复申请页面应有的功能，页面预览效果如图 10.140 所示。

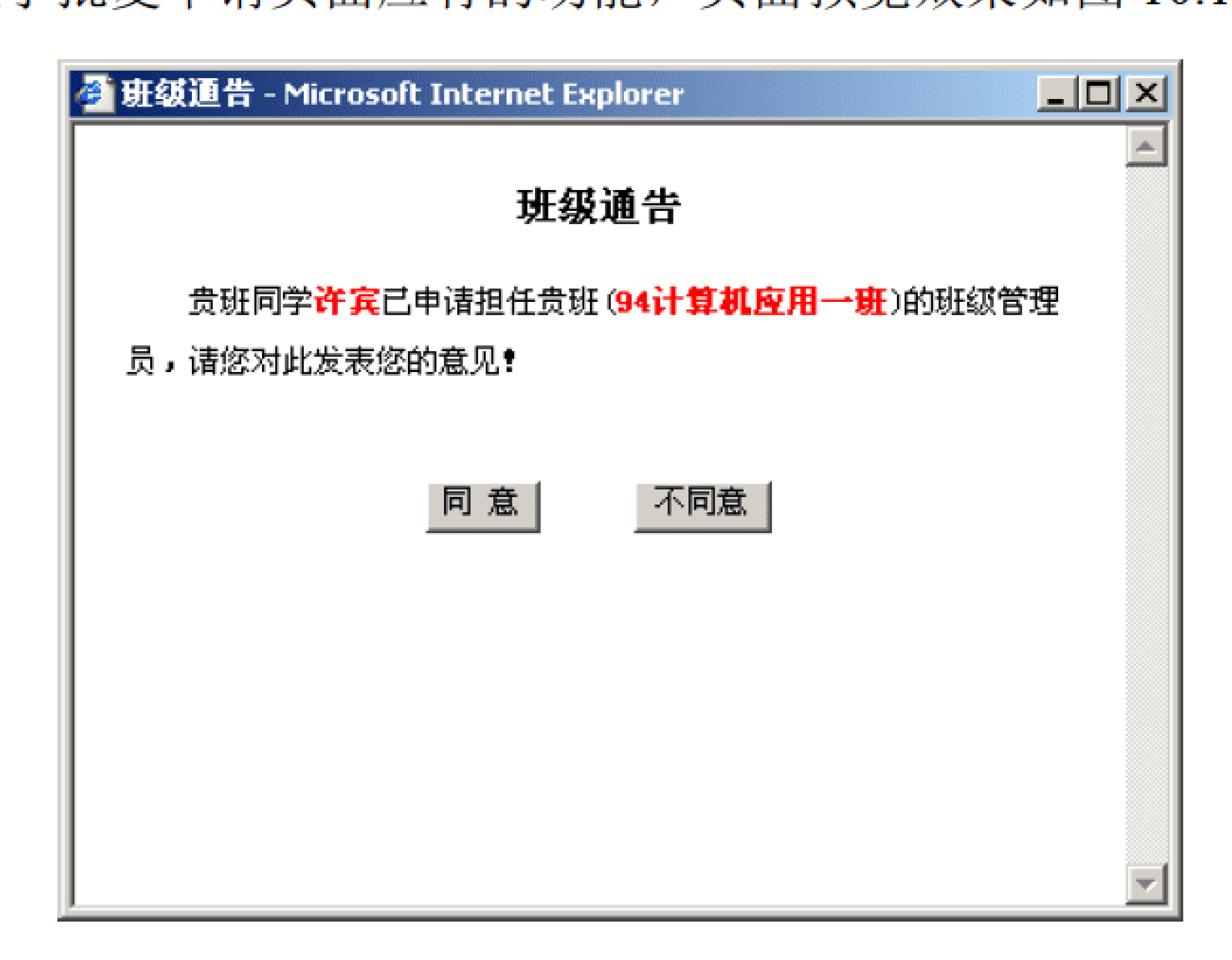


图 10.140 页面预览效果

10.8　搜索学校/班级

在同学录系统中，对学校的搜索和班级的搜索是在同一页面中实现的。事实上，要搜索班级，首先必须找到其所在的学校，然后在该学校的班级列表中才能寻找相应的班级。

10.8.1　快速搜索

新建一个 ASP.NET VB 类型的页面，将其命名为 Search.aspx。选择【修改】|【模板】|【套用模板到页】命令，在弹出的【选择模板】对话框中选择【页面模板】。单击【确定】按钮，将模板应用到本页面。

(1) 在表单 Form 中，插入一个 3 行 3 列的表格 table2，其属性设置如图 10.141 所示。

图 10.141　表格属性

(2) 分别合并表格第 1 行和第 3 行的 3 个单元格，并分别设置其背景图片为 Image 目录下的 lybg_1.gif 和 lybg_2.gif。然后，分别设置表格第 2 行的第 1 个单元格和第 3 个单元格的背景图片为 Image 目录下的 lybg_3.gif 和 lybg_4.gif。

(3) 在表格第 2 行的第 2 个单元格中再次插入一个 5 行 1 列的表格 table3，该表格用于布局用来设置相关搜索条件的文本框和下拉列表框控件，其表格设计如图 10.142 所示。

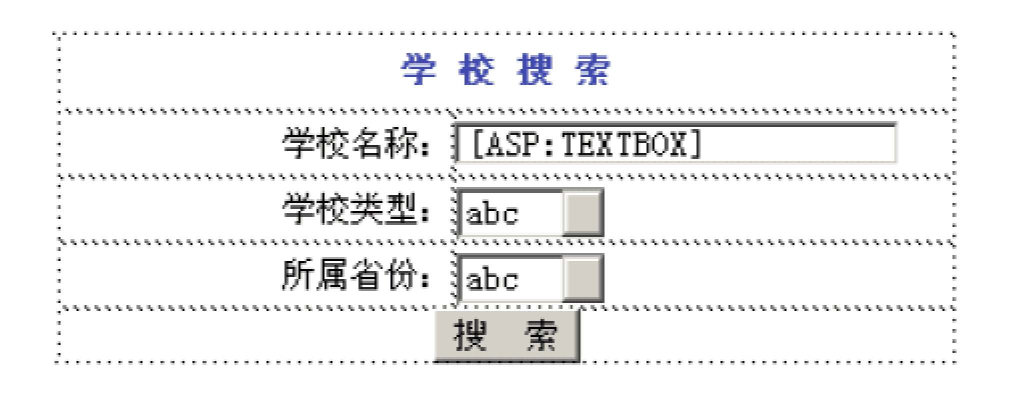


图 10.142　表格设计

在此表格中，插入了两个下拉列表框(DropDownList)控件，分别用于列出学校类型和所属省份，以便用户选择。其中，学校类型下拉列表框中的选项包括“大学”、“中专”、“中学”以及“小学”，其代码如图 10.143 所示。

```
<td align="left" style="padding-left:10px;">
  <asp:DropDownList ID="TType" runat="server">
    <asp:ListItem Value="大学">大　学</asp:ListItem>
    <asp:ListItem Value="中专">中　专</asp:ListItem>
    <asp:ListItem Value="中学">中　学</asp:ListItem>
    <asp:ListItem Value="小学">小　学</asp:ListItem>
  </asp:DropDownList>
</td>
```

图 10.143　DropDownList 控件设计代码

对于【所属省份】下拉列表框中的数据，这里采用了动态绑定的方式。虽然全国的省份、直辖市是固定的，但其数量较多，采用静态方式直接添加列表项较为麻烦。为此，在数据库中特别添加了一个数据表 SYS_PROVINCE，用于存储全国的省份信息。

在绑定【所属省份】下拉列表框中的数据之前，需要创建一个数据集。

(4) 打开【服务器行为】面板，单击 ＋ 按钮，从弹出的菜单中选择【数据集】命令。在弹出的【数据集】对话框中，设置名称为 DataSet1，连接选择 Cnn，表格选择 SYS_PROVINCE，排序方式设置为按字段 ID 升序排序，如图 10.144 所示。

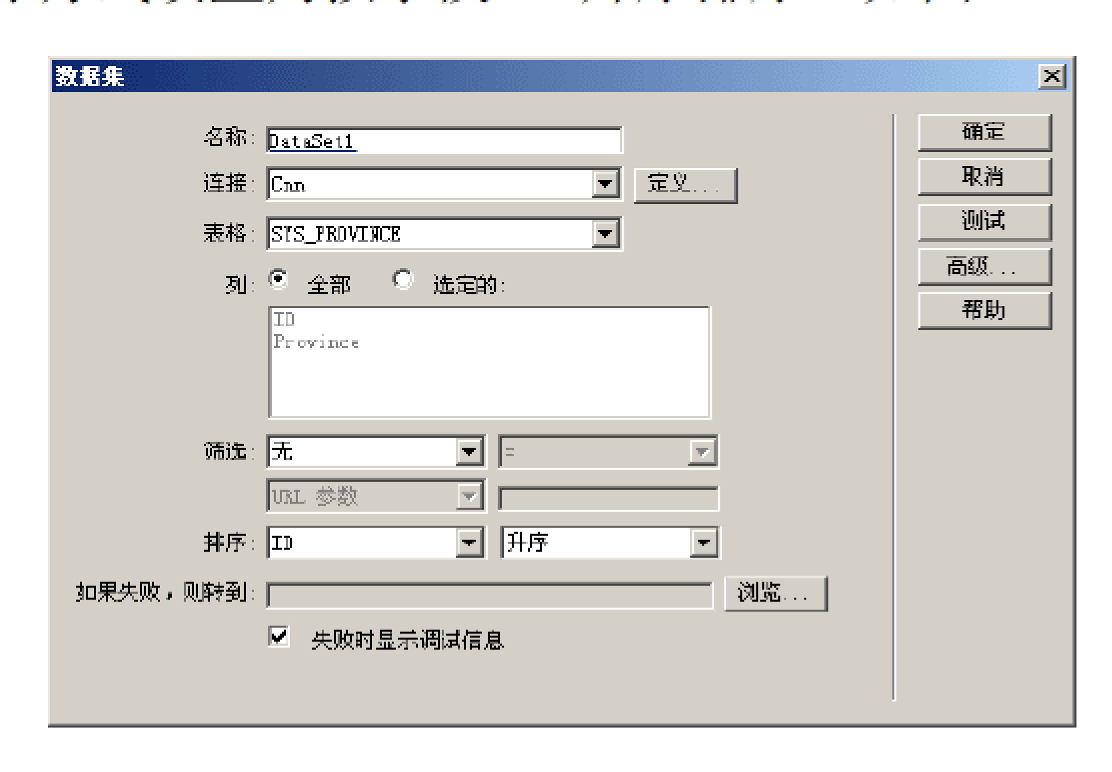


图 10.144　创建数据集 DataSet1

(5) 单击【确定】按钮，完成数据集的创建。在【设计】视图，选择并右击列举省份的下拉列表框控件，在弹出的菜单中选择【编辑标签】命令，此时将弹出【标签编辑器】对话框。选择【数据】分类，如图 10.145 所示。

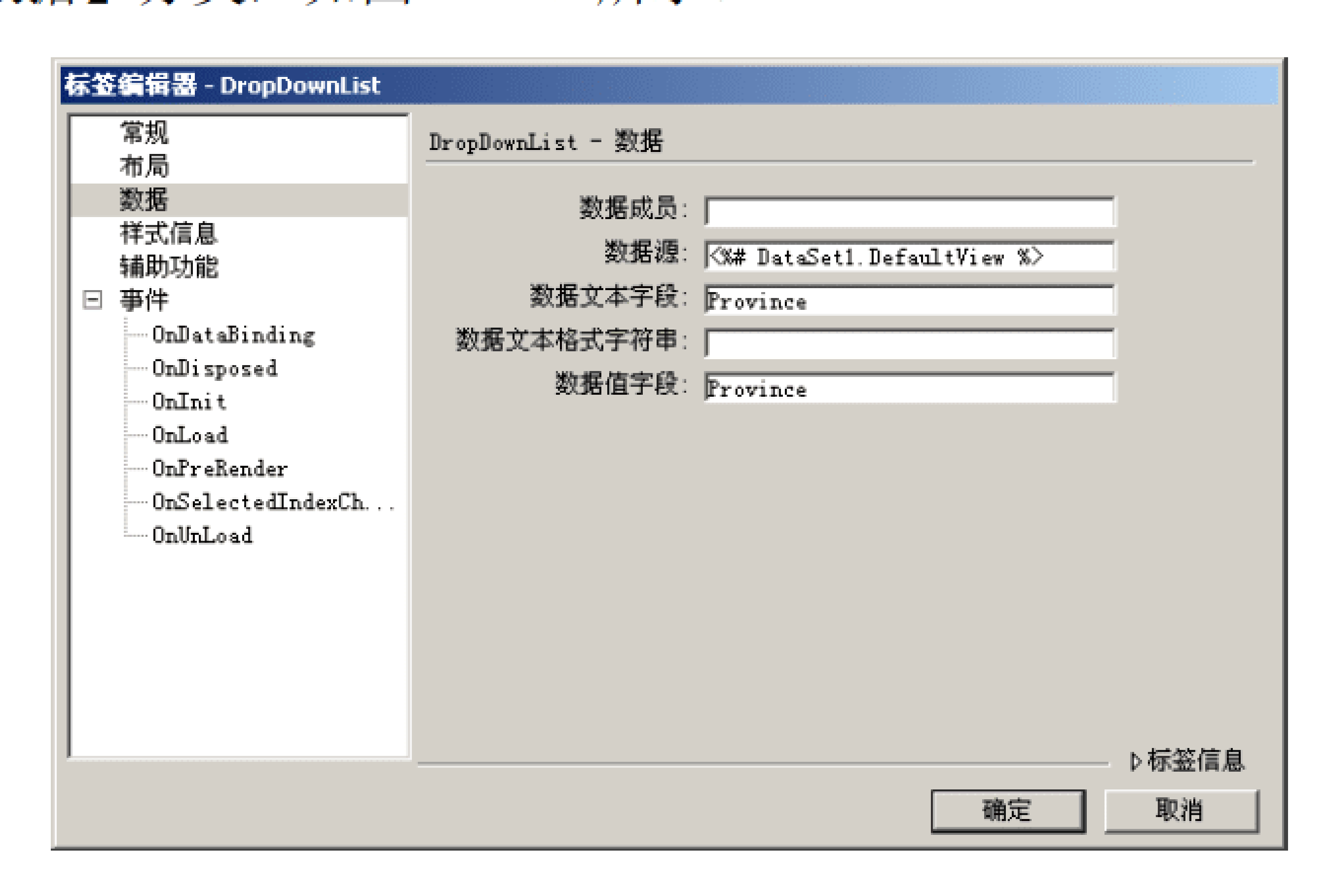


图 10.145　【标签编辑器】对话框

(6) 设置【数据源】为<%# DataSet1.DefaultView %>，【数据文本字段】为 Province，【数据值字段】为 Province。单击【确定】按钮，完成数据绑定的设置。

单击【搜索】按钮，页面将根据用户输入或选择的各项条件对学校信息进行检索，检索结果通过一个记录集来获取。

(7) 在【服务器行为】面板中，再次单击 ＋ 按钮，并在弹出的菜单中选择【数据集】命令。在弹出的【数据集】对话框中，单击【高级】按钮，切换至【数据集】的高级模式。在【数据集】对话框的高级模式中，设置名称为 DataSet2，连接为 Cnn。

由于搜索条件包含 3 个，因此这里必须定义 3 个参数，分别获取 3 个搜索条件的值。

(8) 单击【添加参数】按钮+，在弹出的【编辑参数】对话框中，设置【名称】为 @SchoolName，【类型】为 VarChar，如图 10.146 所示。

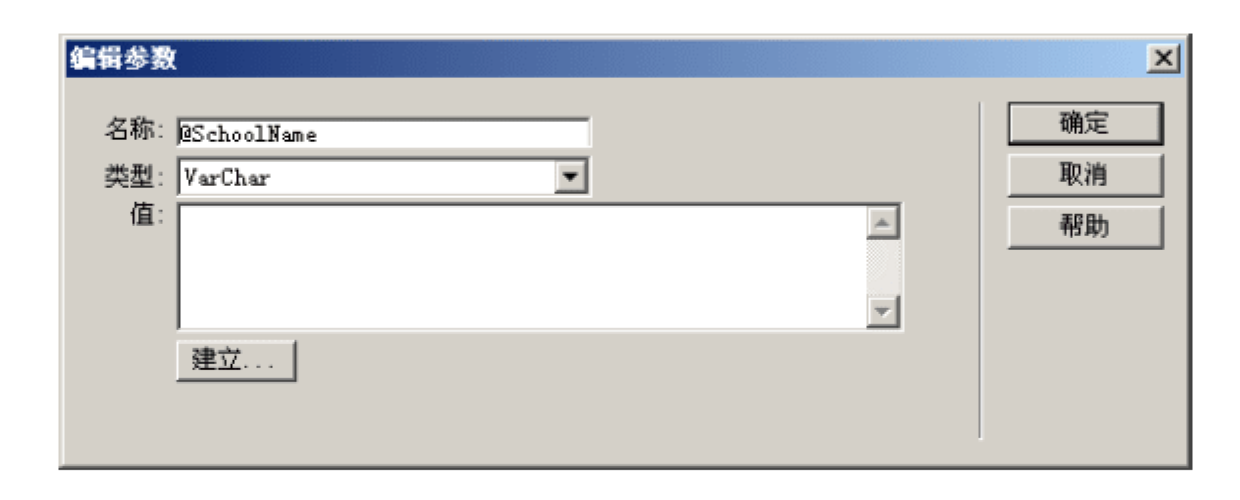


图 10.146　【编辑参数】对话框

(9) 单击【建立】按钮，在弹出的【生成值】对话框中设置【名称】为 SchoolName，【源】为【表单变量】，如图 10.147 所示。

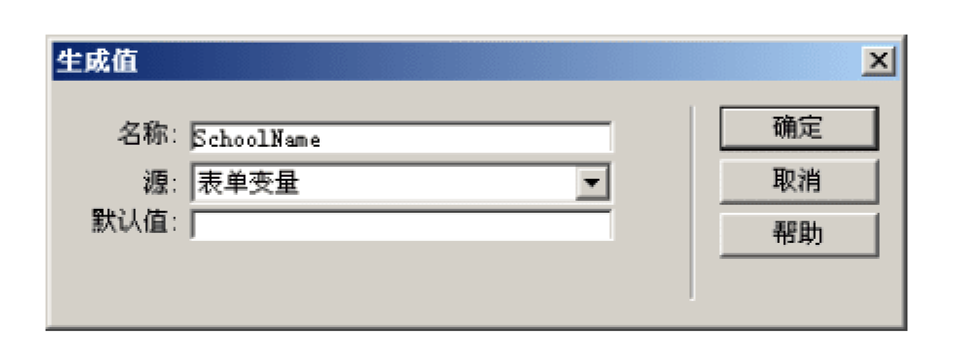


图 10.147　【生成值】对话框

(10) 单击【确定】按钮，完成此参数的设置。重复以上步骤，再次添加两个参数，其参数名分别为@ProVince 和@TType，其参数值分别设置为表单变量 Province 和表单变量 Ttype。

完成参数设置后，在 SQL 文本框中输入以下 SQL 语句：

```
SELECT * FROM schoolinfo WHERE schoolname like '%' + ? + '%' and province=?
and type=? ORDER BY id
```

在该语句中，对学校名称的匹配采用了 like 子语句，即模糊匹配。数据集的设置如图 10.148 所示。

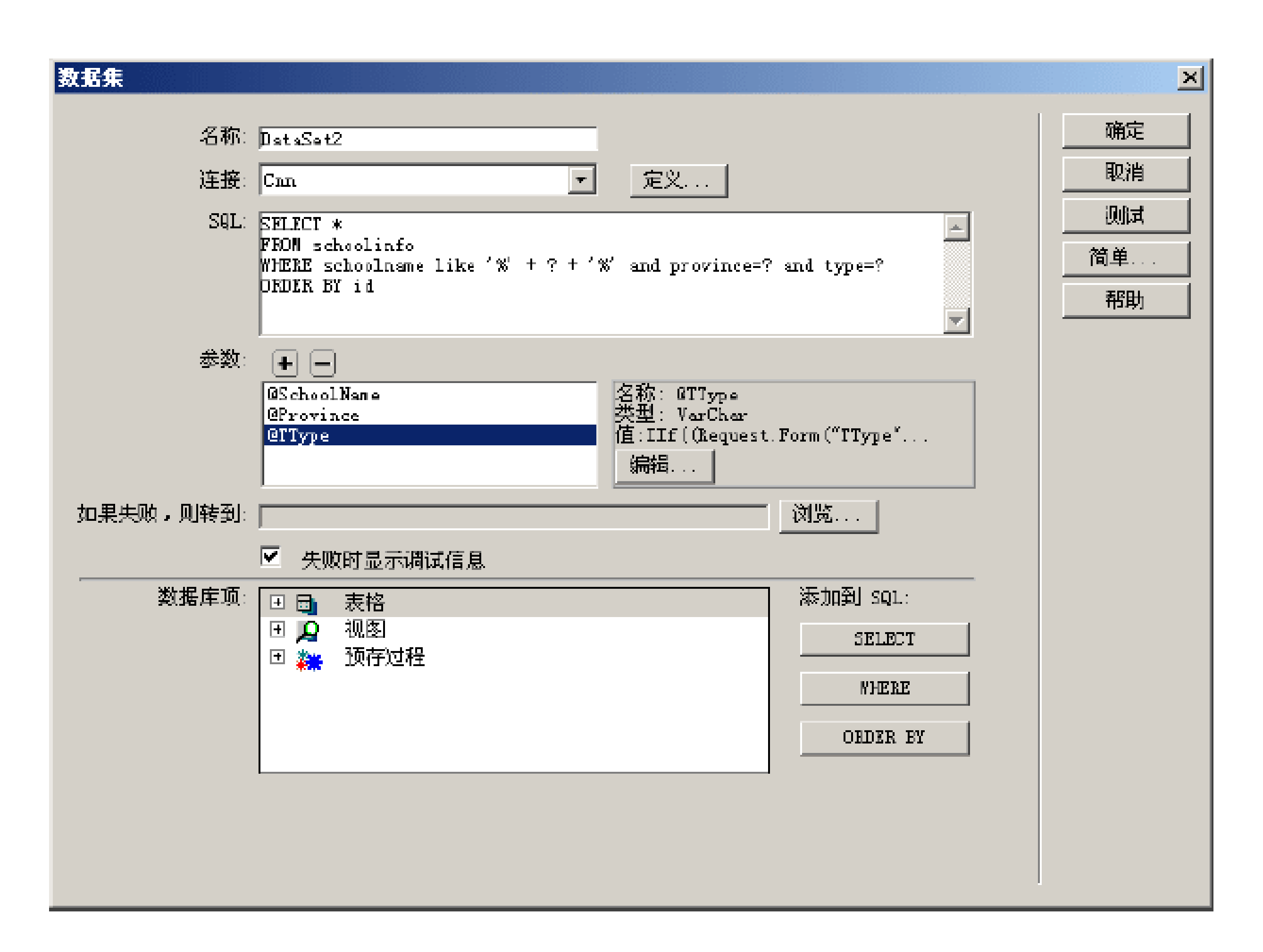


图 10.148　创建数据集 DataSet2

(11) 单击【确定】按钮，完成数据集 DataSet2 的创建，该数据集将返回符合搜索条件的所有学校信息。

接下来，实现搜索结果的显示。考虑到当用户第一次加载此页面时，并不执行搜索操作(即无须显示搜索结果)，因此在搜索结果区域需要添加一个 Panel 控件，用于控制搜索结果区域的显示与否。

(12) 在表格 table2 之后，插入一个 Panel 控件，并设置其 ID 为 Panel1。然后，在【服务器行为】面板中，单击按钮，在弹出的菜单中选择【数据网格】命令。在弹出的【数据网格】对话框中，设置 ID 为 DataGrid1，【数据集】为 DataSet2，【导航】项为【编号链接到每一页】，如图 10.149 所示。

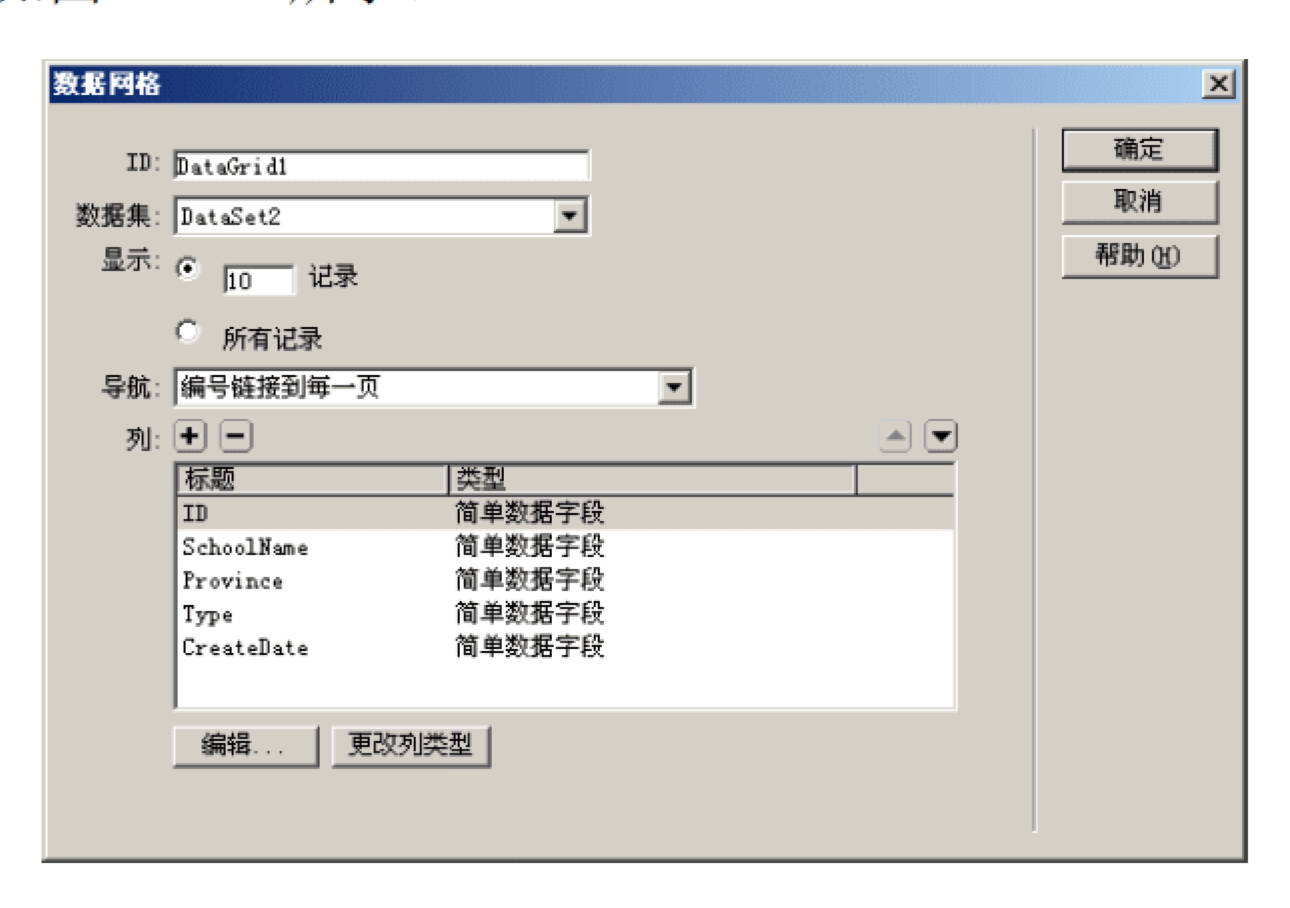


图 10.149　【数据网格】对话框

(13) 在【列】列表框中，选择字段 ID，单击【编辑】按钮，在弹出的【简单数据字段列】对话框中设置【标题】为【编号】，如图 10.150 所示。

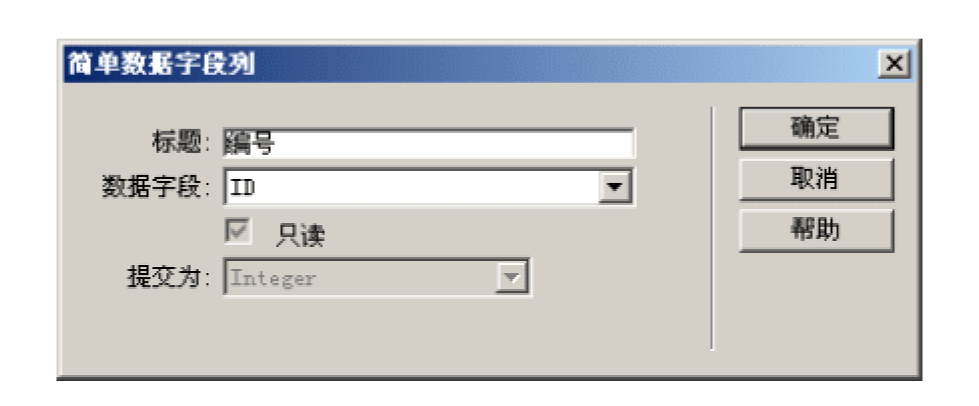


图 10.150　【简单数据字段列】对话框

(14) 单击【确定】按钮，返回【数据网格】对话框。重复以上操作，分别设置字段 SchoolName、Province、Type、CreateDate 的标题名称为“学校名称”、“学校类型”、“所属省份”和“创建时间”。

(15) 为了在显示学校名称的列中创建链接，需要更改该列的类型。选择【学校名称】列，单击【更改列类型】按钮，在弹出的菜单中选择【超级链接】命令，在弹出的【超级链接列】对话框中，设置【超级链接文本】为数据字段 SchoolName，【链接页】为数据字段 ID，【格式字符串】为“School_Info.aspx?id={0}”，如图 10.151 所示。

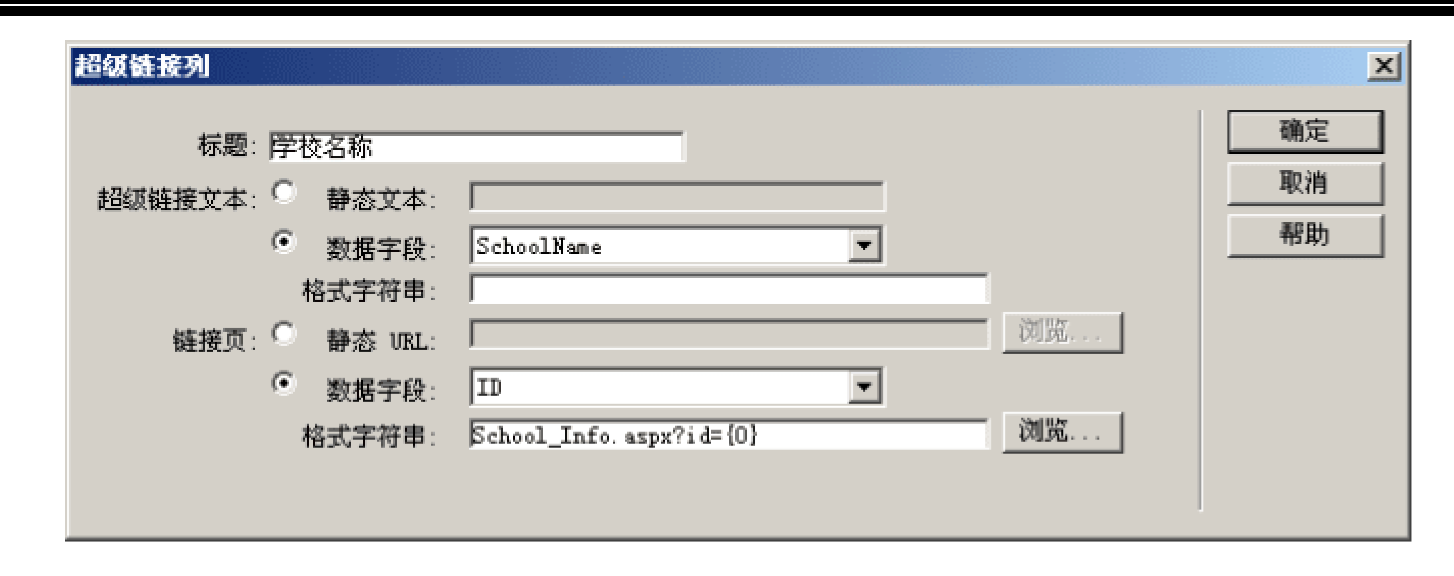


图 10.151　【超级链接列】对话框

(16) 单击【确定】按钮，返回【数据网格】对话框；再次单击【确定】按钮，完成数据网格的创建。

(17) 为了设置搜索结果为空时的提示信息，在定义数据网格之后输入文本“没有符合查询条件的学校信息，请重新搜索！”，并设置其字体颜色为红色。然后选择该文本，在【服务器行为】面板中，单击 + 按钮，从弹出的菜单中选择【显示区域】|【数据集为空时显示】命令。在弹出的【数据集为空时显示】对话框中，选择数据集 DataSet2，如图 10.152 所示。

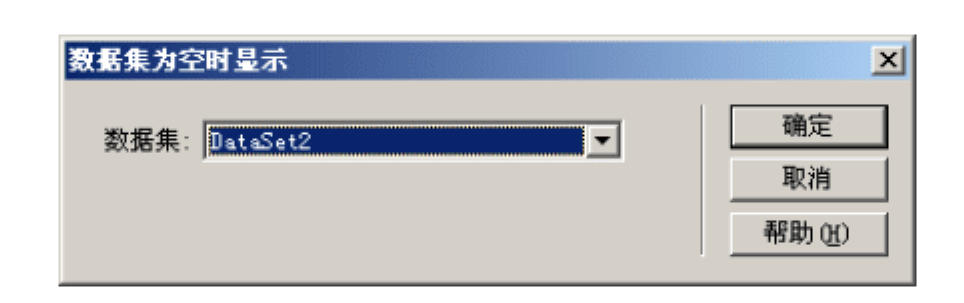


图 10.152　【数据集为空时显示】对话框

单击【确定】按钮，完成该显示区域的设置。

(18) 当用户没有搜索到自己所需要的学校时，应该提供注册新学校的按钮。为此，还需在页面上添加一个按钮控件，其文本为“注册新学校”，同时设置其单击事件为 AddSchool_Click。

切换至【代码】视图，添加【注册新学校】按钮的单击事件，其代码如下：

【示例代码】

```
Sub AddSchool_Click(ByVal Sender As Object, ByVal E As EventArgs)
  '将页面跳转至新增学校功能页面
  Response.Redirect("School_Add.aspx")
End sub
```

该事件仅实现一个链接功能，即将页面跳转至新增学校的功能页面 School_Add.aspx。

(19) 再来看页面加载时所执行的 Page_Load 事件，该事件主要用于控制搜索结果区域的显示与否，其代码如下：

【示例代码】

```
Sub Page_Load(ByVal Sender As Object, ByVal E As EventArgs)
  If Not IsPostBack() Then
    '当页面为初次加载时，不显示搜索结果区域
    Panel1.Visible=False
  Else
    '当页面由数据提交而导致再次加载时，显示搜索结果区域
    panel1.Visible=True
```

```
    End If
End sub
```

至此，页面功能全部完成，页面预览效果如图 10.153 所示。

图 10.153　页面预览效果

当用户设置了相应的搜索条件时，单击【搜索】按钮，即可在下方显示搜索结果，如图 10.154 所示。

图 10.154　搜索结果

10.8.2　查看班级列表

在学校搜索的结果列表中，单击相应的学校名称链接，即可打开新的页面来查看该学校的注册班级信息。

新建一个 ASP.NET VB 类型的页面，将其命名为 School_Info.aspx。选择【修改】|【模板】|【套用模板到页】命令，在弹出的【选择模板】对话框中选择【页面模板】命令。单

击【确定】按钮，将模板应用到本页面。

1. 创建两个数据集

首先，创建两个数据集，分别用于获取当前学校的基本信息及所有班级信息。

(1) 打开【应用程序】面板组，切换至【服务器行为】面板，单击 + 按钮，从弹出的菜单中选择【数据集】命令。在弹出的【数据集】对话框中，设置数据集名称为 DataSet1，连接选择 Cnn，表格选择 SchoolInfo，筛选条件设置为字段 ID 等于 URL 参数 ID，如图 10.155 所示。

图 10.155　创建数据集 DataSet1

(2) 单击【确定】按钮，完成数据集 DataSet1 的创建，该数据集将返回指定班级的基本信息。重复以上操作，再次新增一个数据集。在此数据集中，设置名称为 DataSet2，表格选择 ClassInfo，筛选条件设置为字段 SchoolID 等于 URL 参数 ID。在【列】选项组中，选择【选定的】单选按钮，并在下面的列表框中选择 ClassName、CreateDate、Grade、ID 四个字段(在选中一项后，按住 Ctrl 键，可同时选择其他项)，如图 10.156 所示。

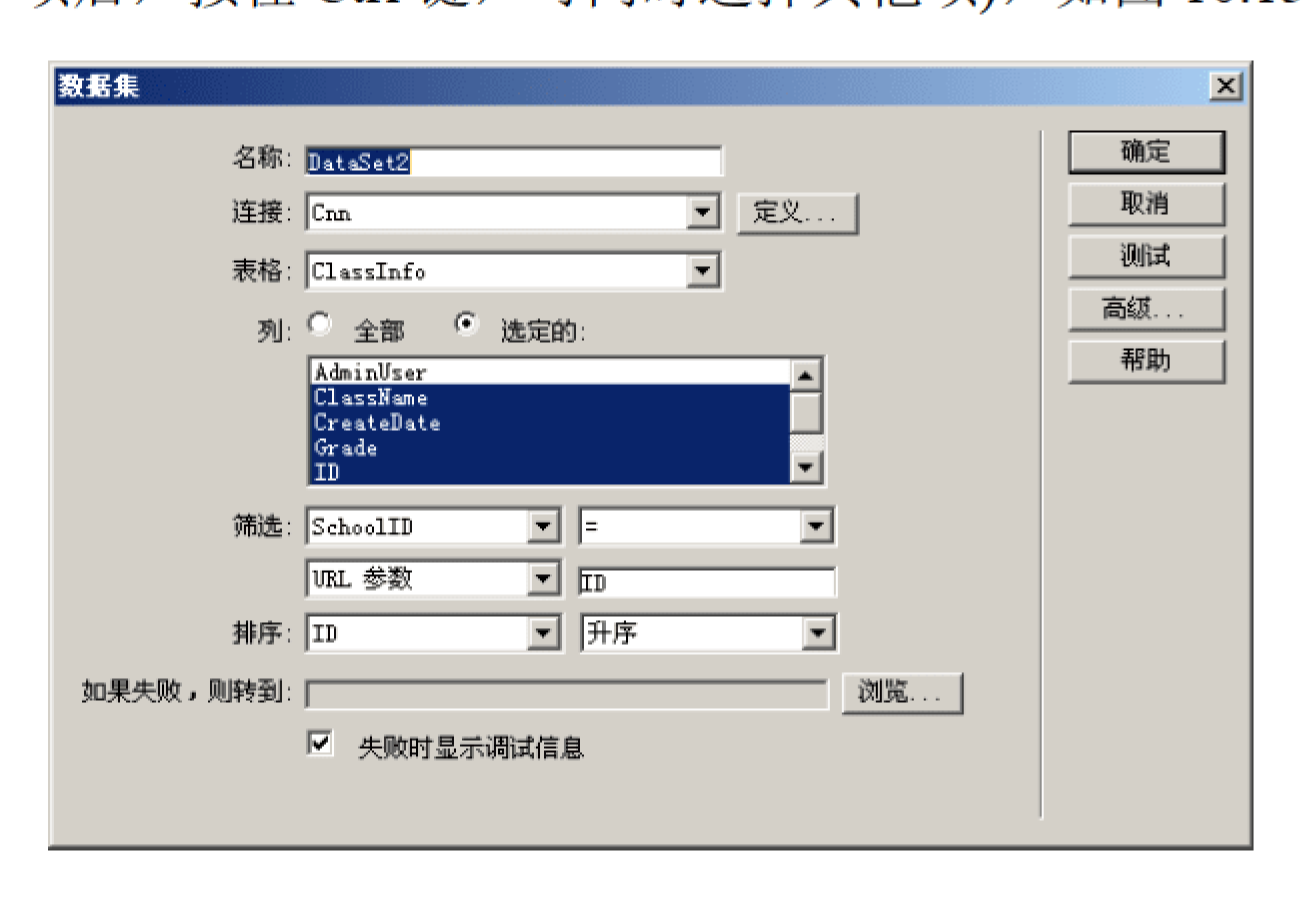


图 10.156　创建数据集 DataSet2

单击【确定】按钮，完成数据集 DataSet2 的创建，该数据集将返回指定学校下的所有班级信息。

(3) 返回页面的【设计】视图，在表单 Form 内输入文本“当前学校：”。然后，在【绑

定】面板中选择并展开数据集 DataSet1，拖动字段 SchoolName 至文本“当前学校：”之后，同时设置其颜色为红色。

(4) 将光标移至上步插入的动态文本之后，按 Enter 键换行。在【服务器行为】面板中，单击+按钮，从弹出的菜单中选择【数据网格】命令。在弹出的【数据网格】对话框中设置 ID 为 DataGrid1，数据集选择 DataSet2，设置一次显示 10 条记录，【导航】选项设置为【编号链接到每一页】，如图 10.157 所示。

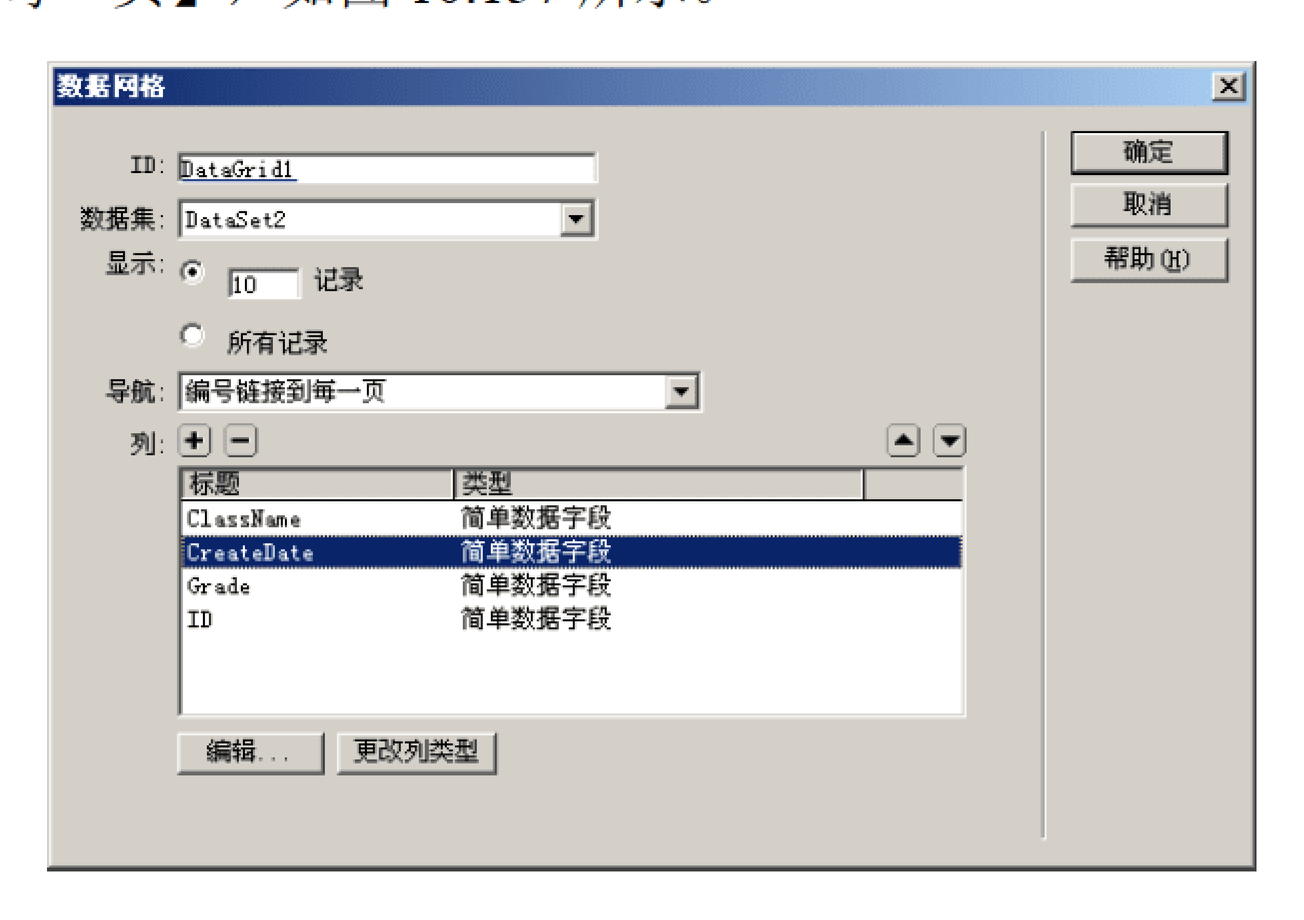


图 10.157 【数据网格】对话框

(5) 这里，字段在【列】列表框中的顺序与在数据网格中所要显示的顺序稍有出入，单击▲按钮和▼按钮，可调整选择列的位置。调整后的列按从上到下的排列，其顺序为 ID、ClassName、Grade、CreateDate。

(6) 选中字段 ID，单击【编辑】按钮，在弹出的【简单数据字段列】对话框中设置【标题】为“班级编号”，如图 10.158 所示。

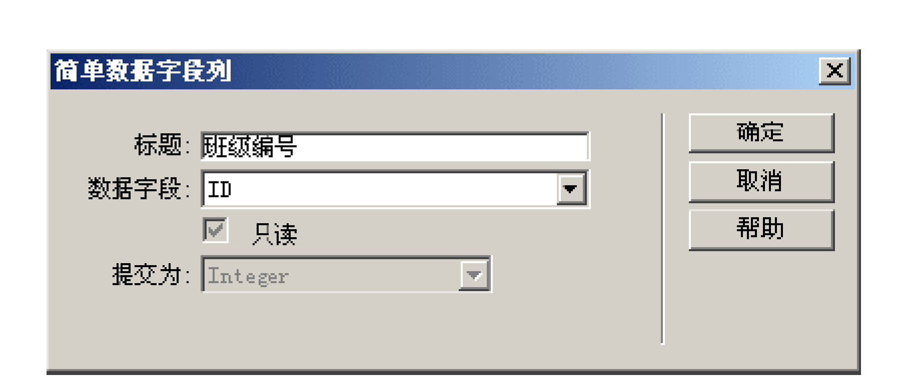


图 10.158 【简单数据字段列】对话框

(7) 重复以上操作，分别设置字段 ClassName、Grade、CreateDate 的标题为“班级名称”、“年级”和“创建时间”。单击【添加列】按钮+，从弹出的菜单中选择【删除按钮】命令，在弹出的【删除按钮列】对话框中设置【标题】为“加入该班级”，【按钮类型】为【链接按钮】，如图 10.159 所示。

图 10.159 【删除按钮列】对话框

单击【确定】按钮，完成该列的添加。

前面提到过，虽然使用的是【删除按钮】命令，但所执行的并不是删除操作，而是自定义的加入班级操作。之所以选择【删除按钮】命令，只是借用这种便捷方式来快速添加一个按钮列。随后，将会修改默认的删除事件为自定义的事件。这种创建按钮列的方法并不值得推荐，虽简捷，但却不易让人理解，甚至会造成误操作(如果没有修改默认的删除事件，系统将会执行删除操作)。事实上，用户也可直接修改【数据网格】服务器行为面板对应的代码，来添加一个按钮列，这才是值得推荐的方法。

在【数据网格】对话框中，单击【确定】按钮，完成数据网格的创建。

(8) 在【设计】视图中，选择并右击数据网格 DataGrid1，从弹出的快捷菜单中选择【编辑标签】命令。在弹出的【标签编辑器】对话框中选择【布局】分类，在右边的列表框中设置宽度值为 80%；同时，选择【事件】分类下的 OnItemCommand，并设置其事件名为 DataGrid1_Mod。单击【确定】按钮，完成对数据网格的设置。

(9) 切换至【代码】视图，添加当单击数据网格中的【加入该班级】链接按钮时所触发的事件 DataGrid1_Mod，其代码如下：

【示例代码】

```
Sub DataGrid1_Mod(ByVal Sender As Object, ByVal E As
DataGridCommandEventArgs)
  '获取所要加入的班级 ID
  Dim codestr As String = E.Item.Cells(0).Text
  Dim Cnn As OleDbConnection
  Dim Cmd As OleDbCommand
  Dim DataR As OleDbDataReader
  Dim StrCnn As String
  Dim Sql As String
  If CType(e.CommandSource, LinkButton).CommandName = "Delete" Then
    '获取数据库连接字符串
    StrCnn =
System.Configuration.ConfigurationSettings.AppSettings("MM_CONNECTION_ST
RING_Cnn")
    '创建数据库连接
    Cnn = New OleDbConnection(StrCnn)
    Cnn.Open()
    '查询当前用户是否已是所要加入的班级中的成员
    Sql = "select * from userclass where usercode=" &
trim(request.Cookies("id").value) & " and classcode=" & trim(CodeStr)
    Cmd = New OleDbCommand(Sql, Cnn)
    DataR=Cmd.ExecuteReader
    if DataR.Read() Then
      '当前用户已是所要加入的班级中的成员，提示相关信息
      Cnn.Close()
      ClientScript.RegisterStartupScript(Me.GetType(), "", GetInfo("您已经
是该班级成员，无须再次加入！"))
    else
      '当前用户不是所要加入的班级中的成员，建立当前用户与指定班级的关联
      DataR.Close()
      '向数据表 UserClass 中插入一条记录
```

```
        Sql="Insert into userclass(usercode,classcode) Values(" &
    trim(request.Cookies("id").value) & "," & trim(CodeStr) & ")"
        Cmd=New OleDbCommand(Sql,Cnn)
        Cmd.ExecuteNonquery
        Cnn.Close()
        ClientScript.RegisterStartupScript(Me.GetType(), "", GetInfo("加入成
    功! "))
      End If
    End If
End Sub
```

最后，还需添加一个【注册新班级】按钮，该按钮主要用于在当前指定的学校下新增一个班级信息。按钮所定义的单击事件为 AddClass_Click，所执行的操作是将页面跳转至新增班级页面 Class_Add.aspx，其代码如下：

【示例代码】

```
Sub AddClass_Click(ByVal Sender As Object, ByVal E As EventArgs)
 '将页面跳转至新增班级的功能页面，同时将当前的学校 ID 作为其页面参数进行传递
 Response.Redirect("Class_Add.aspx?SchoolID=" &
request.QueryString("id"))
End sub
```

至此，页面功能全部完成，页面预览效果如图 10.160 所示。

图 10.160　页面预览效果

10.9　新 增 信 息

此类页面主要包括新增学校和新增班级两个页面。

10.9.1　新增学校

在新增学校页面中，主要实现了新学校的注册功能。

(1) 新建一个 ASP.NET VB 类型的页面，将其命名为 School_Add.aspx。选择【修改】|【模板】|【套用模板到页】命令，在弹出的【选择模板】对话框中选择【页面模板】命令。单击【确定】按钮，将模板应用到本页面。

(2) 由于新增学校页面的布局与搜索学校页面中的搜索条件部分的布局较为相似，因此可将搜索学校页面的部分界面设计代码直接拷贝过来。选择并复制搜索学校页面中的表格 Table2，然后粘贴至本页面所定义的表单 Form 中。

(3) 调整表格 Table2 和嵌套表格 Table3 的样式，及相应的文本，如图 10.161 所示。

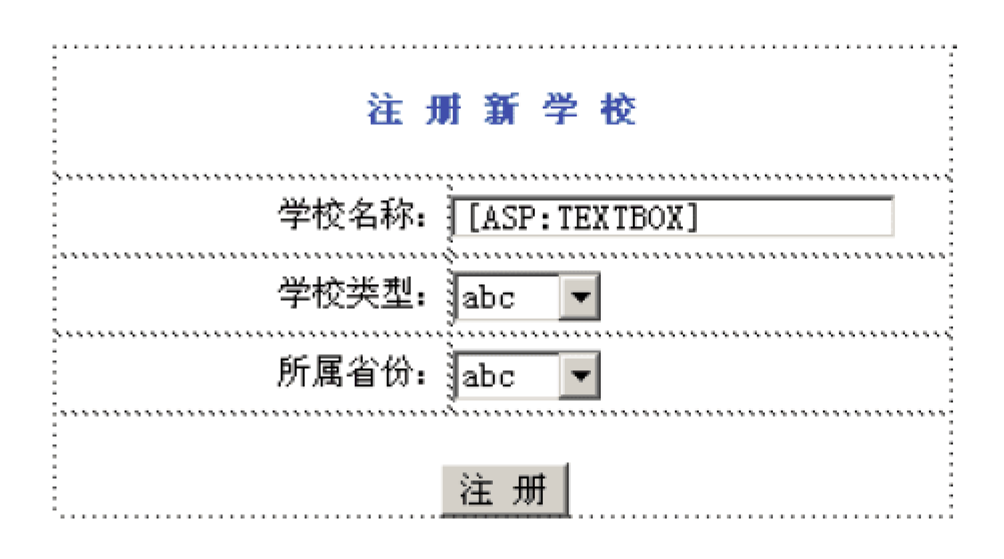


图 10.161　表格设计

(4) 为了绑定【所属省份】下拉列表框的数据显示，仍需创建一个数据集 DataSet1。由于数据集不能跨页面复制，因此只能新建，其数据集的设置与搜索学校页面中的数据集 DataSet1 完全一样，如图 10.162 所示。

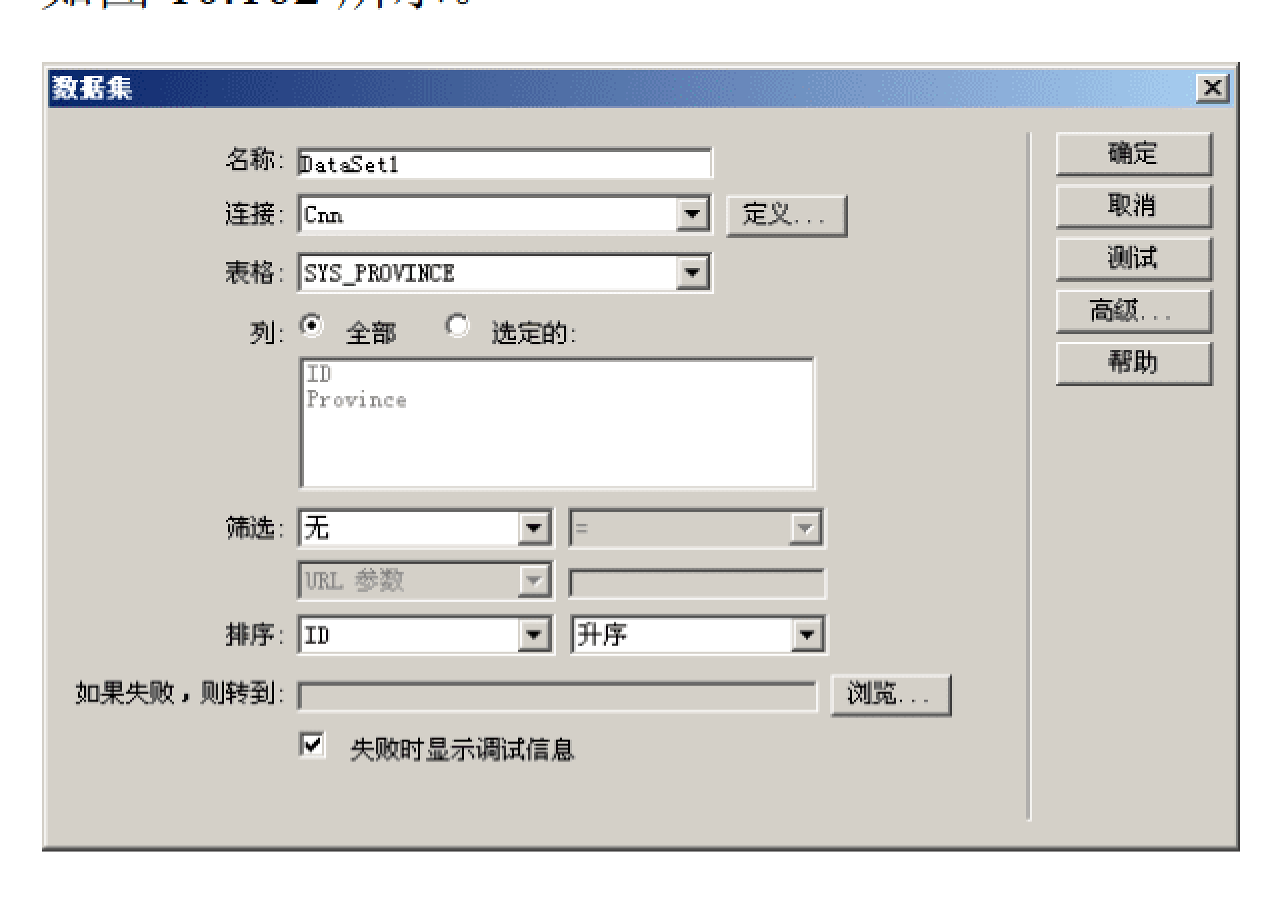


图 10.162　数据集 DataSet1

为了实现数据的新增，需要创建一个【插入记录】服务器行为。

(5) 打开【服务器行为】面板，单击 + 按钮，从弹出的菜单中选择【插入记录】命令。在弹出的【插入记录】对话框中，将【连接】选项设为 Cnn，【插入到表格】选项设为 SchoolInfo，同时设置提交成功后所转向的页面为 School_Add.aspx，如图 10.163 所示。

(6) 在列的设置中，分别设置字段 Province、SchoolName 和 Type 的获取值为表单元素 Province、SchoolNamet 和 Ttype 的取值。

(7) 单击【确定】按钮，完成【插入记录】服务器行为的添加。

至此，新增学校页面的功能完成，页面预览效果如图 10.164 所示。

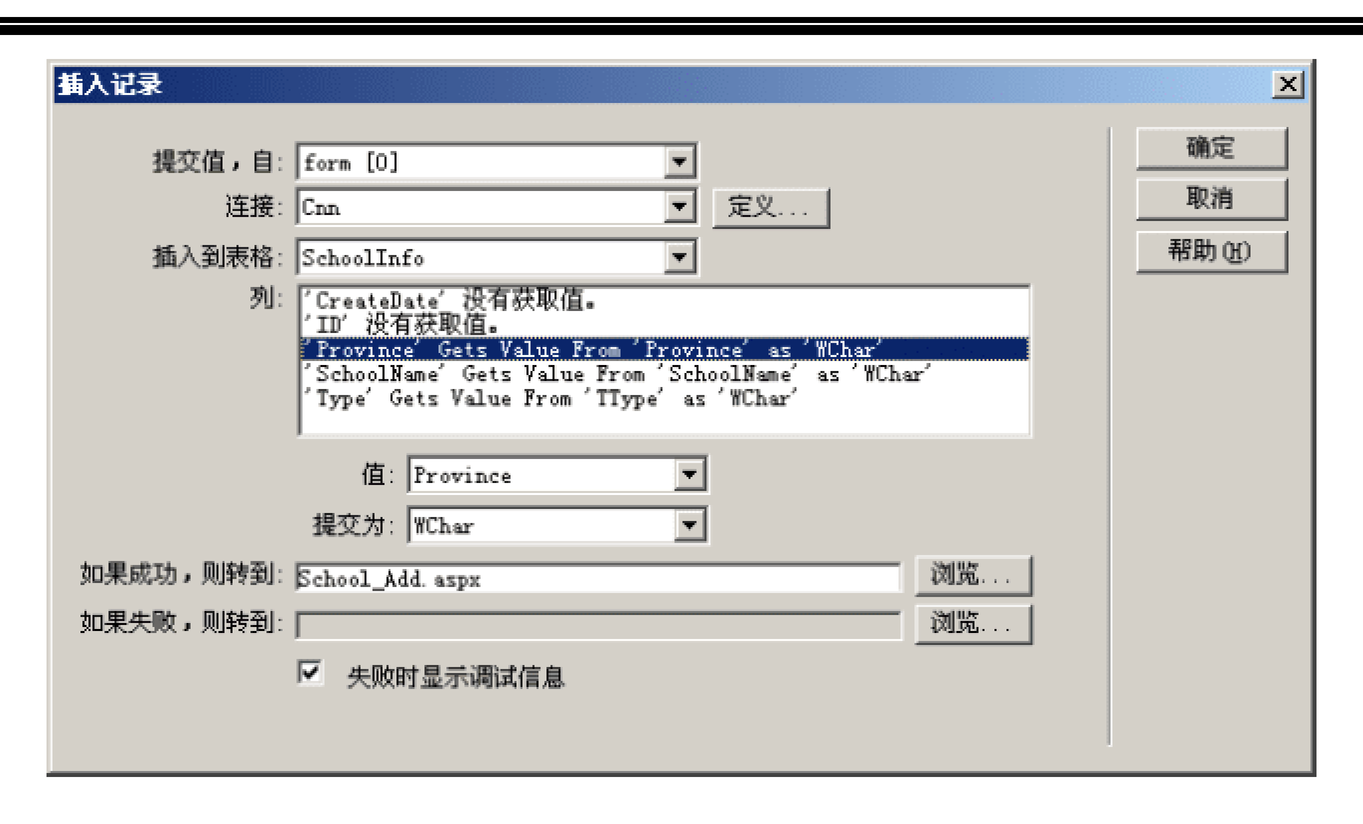


图 10.163　【插入记录】对话框

图 10.164　页面预览效果

10.9.2　新增班级

新增班级页面主要用于在指定的学校下添加新的班级信息。

(1)　新建一个 ASP.NET VB 类型的页面，将其命名为 Class_Add.aspx。选择【修改】|【模板】|【套用模板到页】命令，在弹出的【选择模板】对话框中选择【页面模板】命令。单击【确定】按钮，将模板应用到本页面。

(2)　新增班级页面的布局与新增学校页面的布局较为相似，因此可将新增学校页面中的表格 table2 直接复制过来，并对其嵌套表格 table3 的布局及内容进行修改，修改后的表格布局如图 10.165 所示。

在文本“所属学校”后的单元格中，插入了一个隐藏域，该隐藏域用于存储当前的学校 ID，以便在定义【插入记录】服务器行为时使用。

(3)　在【服务器行为】面板中单击+按钮，从弹出的菜单中选择【数据集】命令。在弹出的【数据集】对话框中设置其【名称】为 DataSet1，【连接】为 Cnn，【表格】为 SchoolInfo，筛选条件设置为字段 ID 等于 URL 参数 SchoolID，如图 10.166 所示。

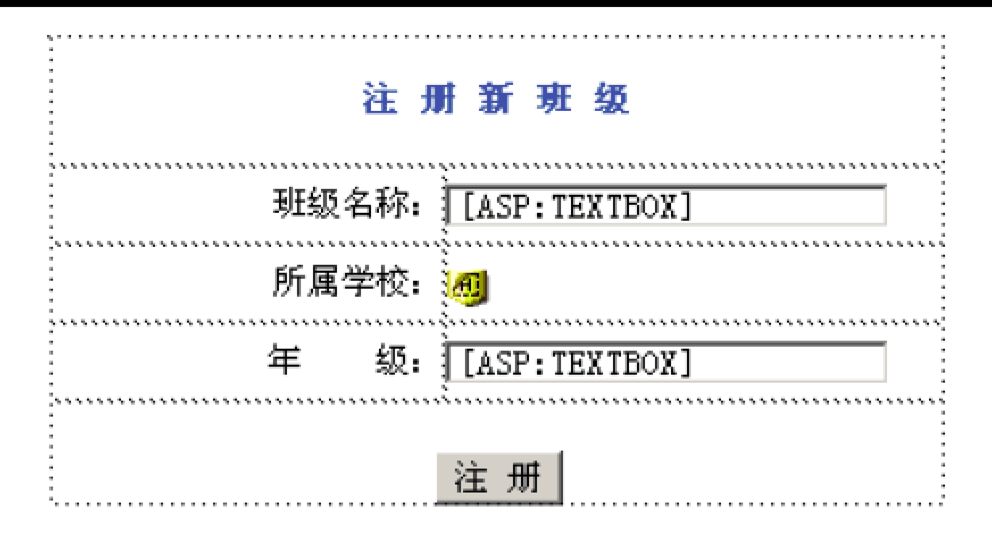


图 10.165　修改后的表格

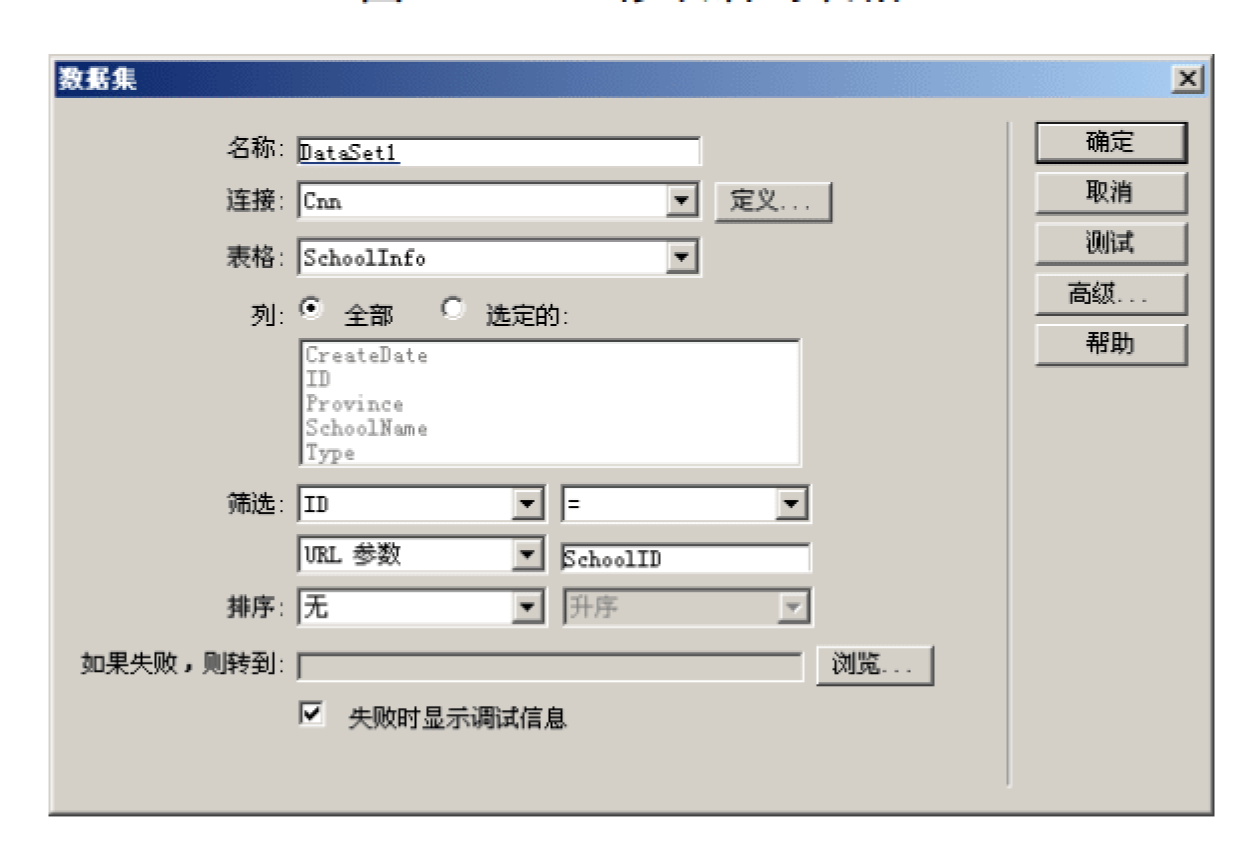


图 10.166　创建数据集 DataSet1

单击【确定】按钮，完成数据集 DataSet1 的创建，该数据集将返回指定 ID 的学校信息。

(4)　在【绑定】面板中，选择并展开数据集 DataSet1，拖动字段 SchoolName 至表格中的隐藏域之前，并设置其字体颜色为红色。

(5)　在【设计】视图中选择隐藏域并右击，从弹出的快捷菜单中选择【编辑标签】命令，在弹出的【标签编辑器】对话框中设置名称为 SchoolID，其值为“<%# DataSet1.FieldValue("id", Container) %>”，如图 10.167 所示。

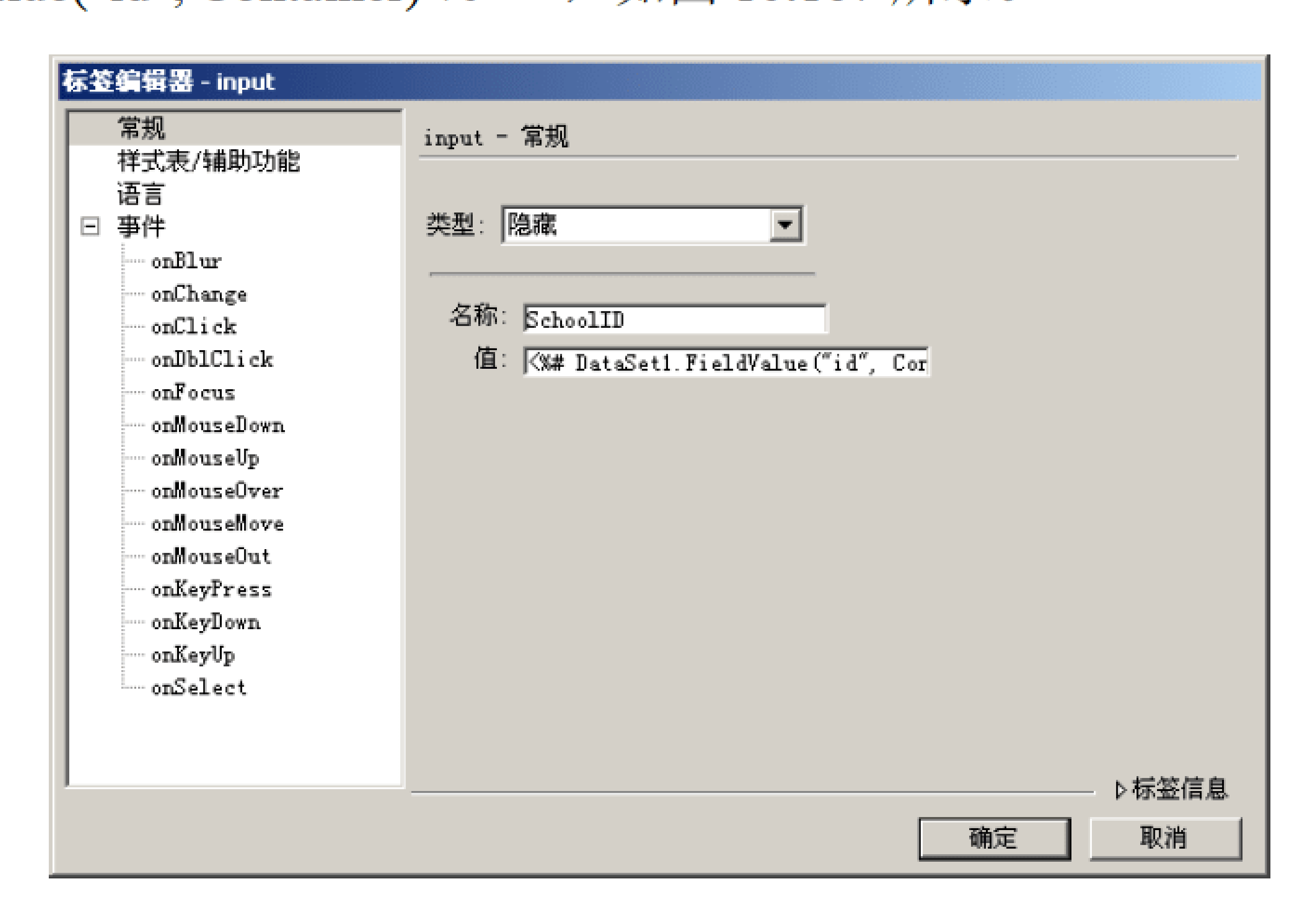


图 10.167　编辑隐藏域

单击【确定】按钮，完成对隐藏域的设置。

(6)　在【应用程序】面板组中，切换至【服务器行为】面板，单击 按钮，从弹出的

菜单中选择【插入记录】命令。在弹出的【插入记录】对话框中，将【连接】选项设为 Cnn，【插入到表格】选项设为 ClassInfo。然后，分别设置字段 ClassName、Grade、SchoolID 的获取值为表单元素 ClassName、Grade 和 SchoolID 的取值。最后，设置数据提交成功后所转向的页面为“School_Info.aspx?id= <%# DataSet1.FieldValue("SchoolID", Container) %>”，如图 10.168 所示。

单击【确定】按钮，完成插入记录的添加。

插入记录
提交值，自: form [0]
连接: Cnn 定义...
插入到表格: ClassInfo
列: 'AdminUser' 没有获取值。
'ClassName' Gets Value From 'ClassName' as 'WChar'
'CreateDate' 没有获取值。
'Grade' Gets Value From 'Grade' as 'Integer'
'ID' 没有获取值。
'SchoolID' Gets Value From 'SchoolID' as 'Integer'
值: SchoolID
提交为: Integer
如果成功，则转到: School_Info.aspx?id=<%# DataSet1.FieldValue("Scho 浏览...
如果失败，则转到: 浏览...
失败时显示调试信息
确定
取消
帮助(H)

图 10.168　【插入记录】对话框

至此，新增班级的页面功能完成，页面预览效果如图 10.169 所示。

当页面提交成功后，系统将自动转向当前学校的班级列表页面。

图 10.169　页面预览效果

10.10　搜 索 同 学

在同学录的主页面上，还提供了一项搜索功能，那就是搜索同学。本节将介绍搜索同学页面功能的具体实现。

(1) 新建一个 ASP.NET VB 类型的页面，将其命名为 Search_Classmate.aspx。选择【修改】|【模板】|【套用模板到页】命令，在弹出的【选择模板】对话框中选择【页面模板】。单击【确定】按钮，将模板应用到本页面。

(2) 选择并复制搜索学校页面中的表格 table2，然后粘贴至本页面。调整其嵌套表格 table3 的布局，如图 10.170 所示。

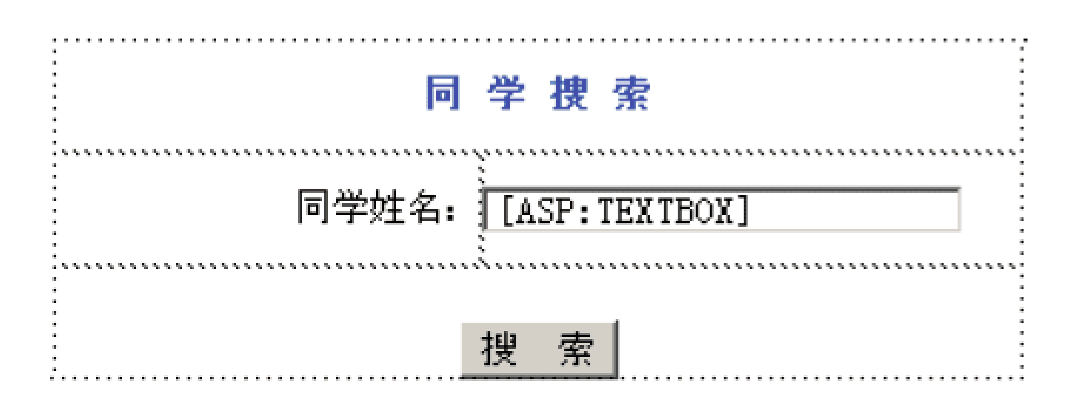


图 10.170　调整字的表格

(3) 打开【服务器行为】面板单击按钮，从弹出的菜单中选择【数据集】命令。在弹出的【数据集】对话框中，单击【高级】按钮，切换至【数据集】对话框的高级模式。在【数据集】对话框的高级模式中，设置【名称】为 DataSet1，【连接】为 Cnn。

(4) 为了匹配用户所输入的查询条件，需要创建一个参数。单击【添加参数】按钮，在弹出的【编辑参数】对话框中，设置【名称】为“@TxName”，【类型】为 VarChar，如图 10.171 所示。

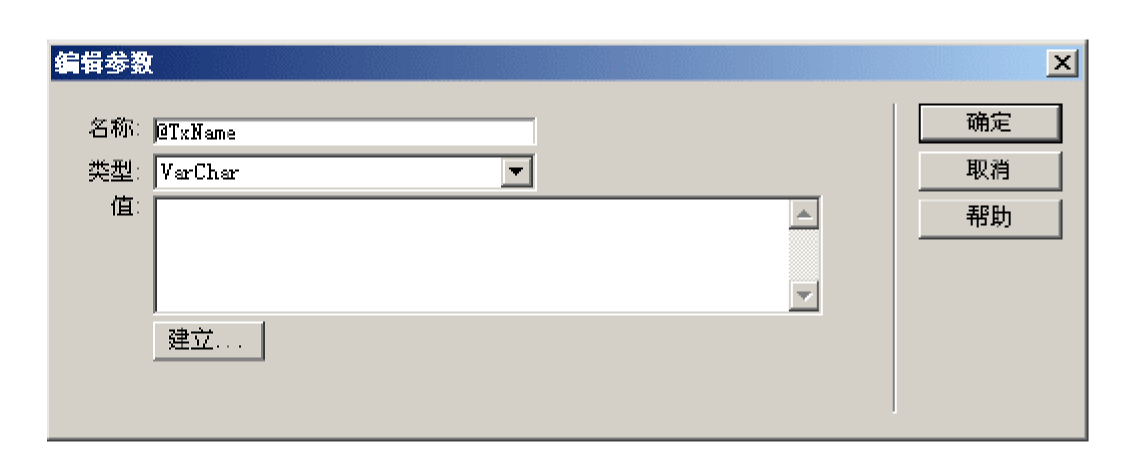


图 10.171　【编辑参数】对话框

(5) 单击【建立】按钮，在弹出的【生成值】对话框中设置【名称】为 TxName，【源】为【表单变量】，如图 10.172 所示。

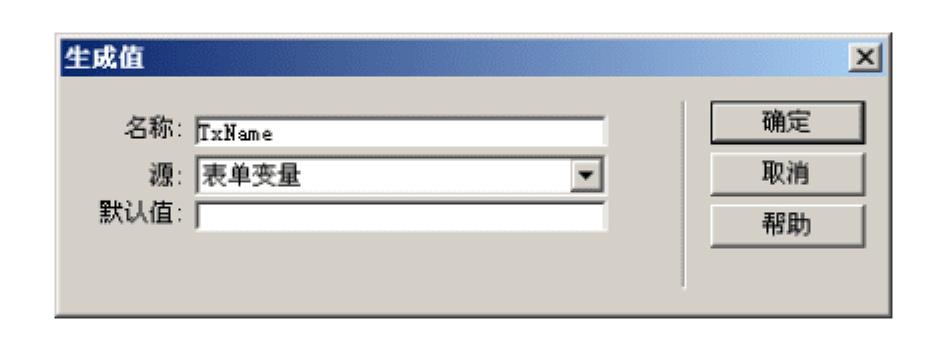


图 10.172　【生成值】对话框

(6) 单击【确定】按钮，完成此参数的设置。然后，在 SQL 文本框中输入以下 SQL 语句：

```
SELECT
sersheet.code,username,schoolname,classname,sex,tell,email,address FROM
usersheet,userclass,classinfo,schoolinfo WHERE usersheet.code=usercode
and classcode=classinfo.id and schoolid=schoolinfo.id and username like '%'
+ ? + '%' ORDER BY code
```

在该语句中，对同学姓名的匹配采用了 like 子语句，即模糊匹配，如图 10.173 所示。

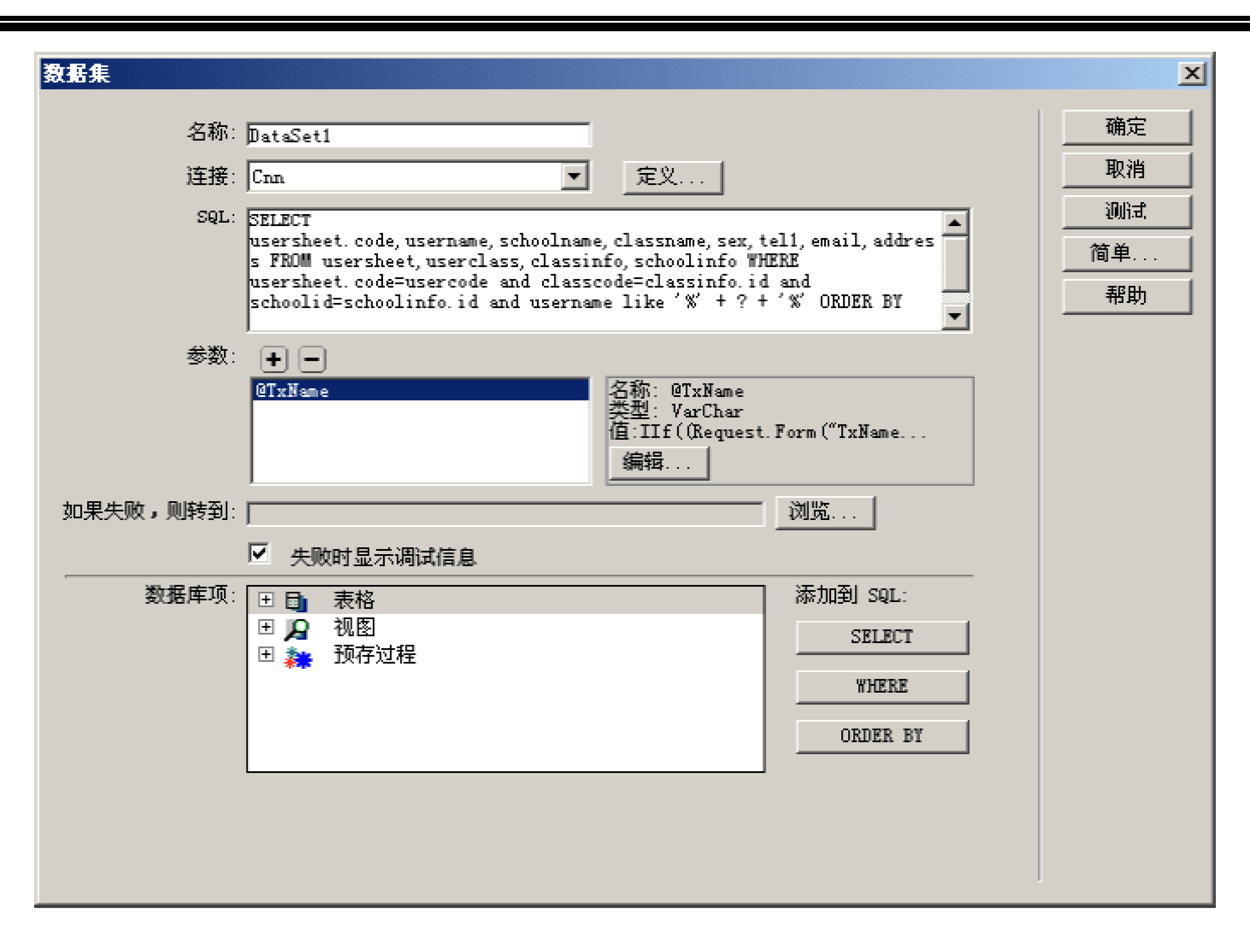


图 10.173　创建数据集 DataSet1

单击【确定】按钮，完成数据集 DataSet1 的创建，该数据集将返回符合搜索条件的所有同学信息。

(7)　在表格 table2 之后，插入一个 Panel 控件(用于控制搜索结果区域的显示与否)，并设置其 ID 为 Panel1。然后，在【服务器行为】面板中单击按钮，从弹出的菜单中选择【数据网格】命令。在弹出的【数据网格】对话框中，设置 ID 为 DataGrid1，【数据集】为 DataSet1，【导航】为【编号链接到每一页】，然后依次设置各列所对应的标题，如图 10.174 所示。

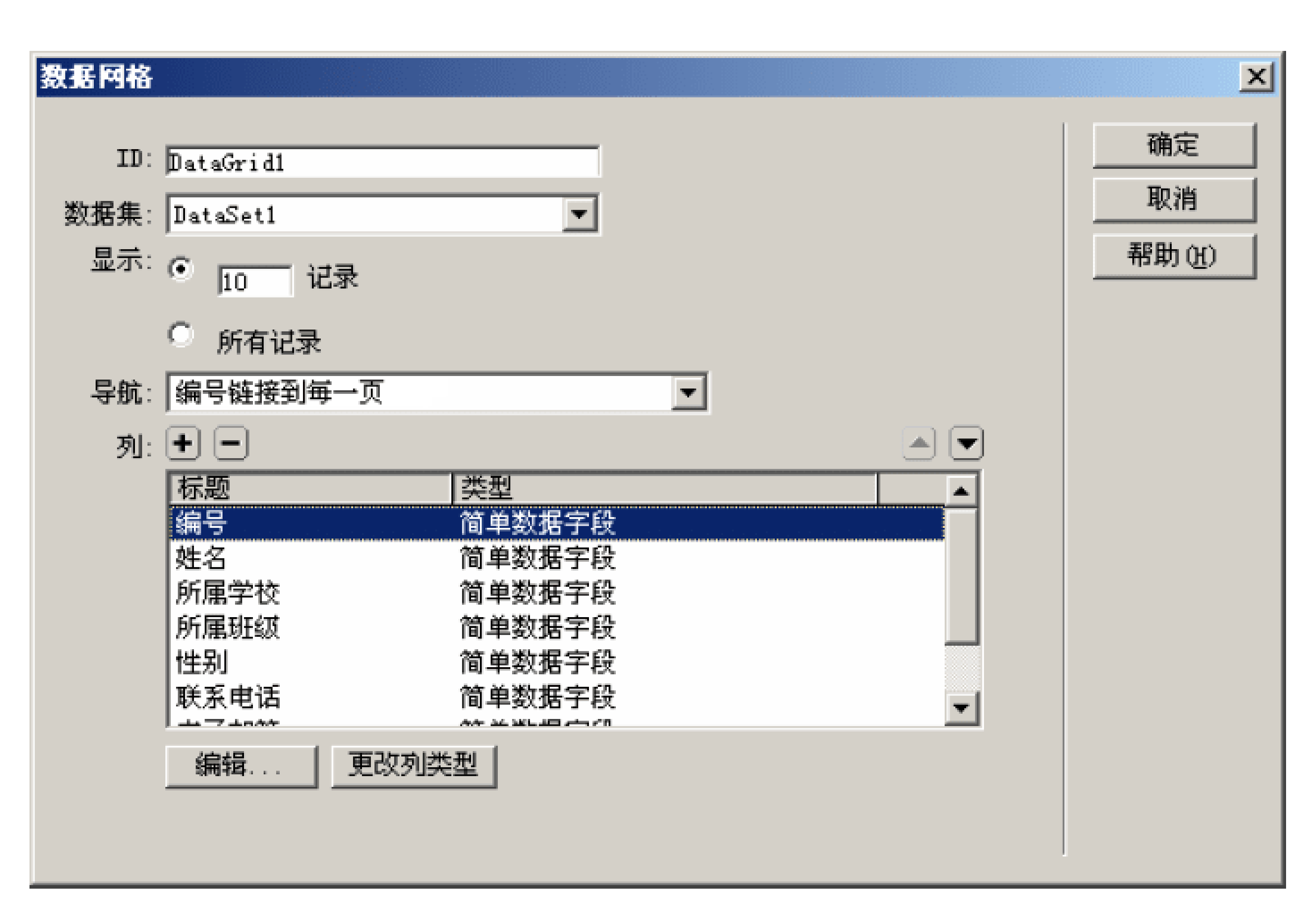


图 10.174　【数据网格】对话框

单击【确定】按钮，完成数据网格的创建。

(8)　在所定义数据网格之后输入文本“没有符合查询条件的同学信息，请重新搜索！”，并设置其字体颜色为红色。然后，选择该文本，在【服务器行为】面板中单击按钮，从弹出的菜单中选择【显示区域】|【数据集为空时显示】命令。在弹出的【数据集为空时显示】对话框中，选择数据集 DataSet1，如图 10.175 所示。

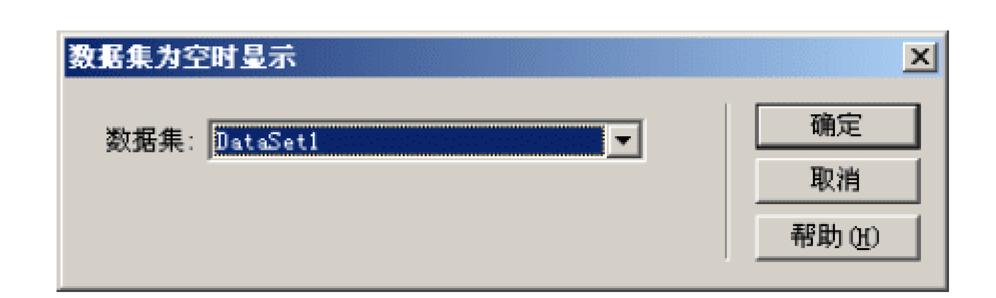


图 10.175　【数据集为空时显示】对话框

单击【确定】按钮，完成该显示区域的设置。

(9)　切换至【代码】视图，添加页面加载时所执行的 Page_Load 事件，其代码如下：

【示例代码】

```
Sub Page_Load(ByVal Sender As Object, ByVal E As EventArgs)
  If Not IsPostBack() Then
    '当页面为初次加载时，不显示搜索结果区域
    Panel1.Visible=False
  Else
    '当页面由数据提交而导致再次加载时，显示搜索结果区域
    panel1.Visible=True
  End If
End sub
```

在该事件中，根据 Page 对象的 IsPostBack 属性来判断页面是否为初次加载，并依此控制搜索结果区域的显示与否。

至此，页面功能全部完成，页面预览效果如图 10.176 所示。

图 10.176　页面预览效果

输入所要搜索的同学姓名，单击【搜索】按钮，即可在下方显示搜索结果，如图 10.177 所示。

编号	姓名	所属学校	所属班级	性别	联系电话	电子邮箱	联系地址
1	胡勇辉	浏河文化补习学校	浏河文化补习学校理三班	男	5591037	hyhlxt@sina.com	湖南省长沙市
1	胡勇辉	重庆建筑大学	西安电子科技大学	男	5591037	hyhlxt@sina.com	湖南省长沙市
1	胡勇辉	浏阳市第一中学	浏阳一中	男	5591037	hyhlxt@sina.com	湖南省长沙市
1	胡勇辉	重庆大学	94计算机应用一班	男	5591037	hyhlxt@sina.com	湖南省长沙市

图 10.177　搜索结果

10.11　习　　题

(1)　在本章实例的基础上，添加一个后台管理页面，允许对省份参数信息进行设置，包括新增、修改和删除。

(2)　添加对班级留言和相片评论的管理功能。若当前用户为班级管理员时，允许该用户对该班级所发表的班级留言和相片的评论信息进行删除。

(3)　在班级通讯录中添加一个模块，允许用户添加非本班级成员的其他同学的通讯录信息。

第 11 章　动态新闻发布系统

动态新闻发布系统是大型网站的一个重要组成部分，它可以提供对网站新闻的动态管理，使之更及时、管理效率更高。

11.1　系 统 分 析

11.1.1　系统功能

1. 新闻浏览

按照新闻分类，列出该分类下的所有新闻的简单信息，包括新闻标题、新闻来源、发布时间等。这些信息被安排在新闻浏览页面。同时，每条新闻的标题将会显示为一个超级链接，单击即可打开一个新的页面来阅读该条新闻。

2. 新闻搜索

在新闻浏览页面中提供了新闻搜索功能。用户可按照新闻分类、关键字或新闻内容进行搜索。在搜索结果中，将列出符合条件的新闻信息，其信息包括新闻标题、所属分类、新闻来源、发布时间等。单击新闻标题，可打开新的页面阅读此条新闻。

3. 新闻阅读

单击新闻浏览页面中的标题链接，即可打开新闻阅读页面。在该页面中，将显示指定新闻的具体内容及相关信息。

4. 新闻管理

对现有的新闻进行管理，包括对指定新闻的内容进行编辑、修改和删除等操作。

5. 新闻发布

添加新闻，其内容包括新闻的所属类、新闻标题、新闻内容、作者、新闻来源、发布时间、关键字、是否通过审核等。

6. 分类管理

对新闻的分类信息进行管理，包括对新闻分类的编辑和删除操作。

7. 添加分类

添加新闻分类，允许在指定的新闻分类下添加子分类。

在以上功能中，新闻浏览、新闻搜索、新闻阅读属系统的前台功能，任何用户均可查看和操作；而新闻管理、新闻发布、分类管理以及添加分类等则属系统的后台功能，仅管

理员可以进行操作。

11.1.2　数据库的建立

本系统使用 Microsoft Access 2000 类型的数据库，其数据库文件名为 Data.Mdb，在该数据库中仅添加了 ClassInfo(新闻分类信息表)和 ArtInfo(新闻信息表)两个数据表。

1. 分类信息表(ClassInfo)

ClassInfo 数据表主要用于存储新闻的分类信息，其表结构见表 11.1。

表 11.1　ClassInfo 数据表

字 段 名	数据类型	字段描述	字段大小	默 认 值	主　键
ClassID	自动编号	新闻分类编号	长整型		P
UpClass	数字	上级分类编号		0	
ClassName	文本	新闻分类名称	50		

2. 新闻信息表(ArtInfo)

ArtInfo 数据表主要用于存储已发布的新闻的基本信息，其表结构见表 11.2。

表 11.2　ArtInfo 数据表

字 段 名	数据类型	字段描述	字段大小	默 认 值	主　键
ID	自动编号	新闻编号	长整型		P
ClassId	数字	所属的新闻分类编号			
ArtTitle	文本	新闻标题	50		
ArtContent	备注	新闻内容			
Author	文本	新闻作者	50		
CopyFrom	文本	新闻来源	50		
Addtime	日期/时间	新闻的发布时间		Now()	
KeyS	文本	用于快速搜索的关键字	50		
ClickNum	数字	点击次数		0	
SH	数字	审核标志		0	

说明：字段 ClassID 表示新闻所属分类的编号，该字段与 ClassInfo 数据表中的字段 ID 相关联。字段 SH 表示新闻的审核标志，其值为 0 表示新闻尚未审核，其值为 1 表示新闻已通过审核，在前台新闻列表中所显示的均是已通过审核的新闻。

11.1.3　站点设置

本例中，站点设置的本地信息如图 11.1 所示。

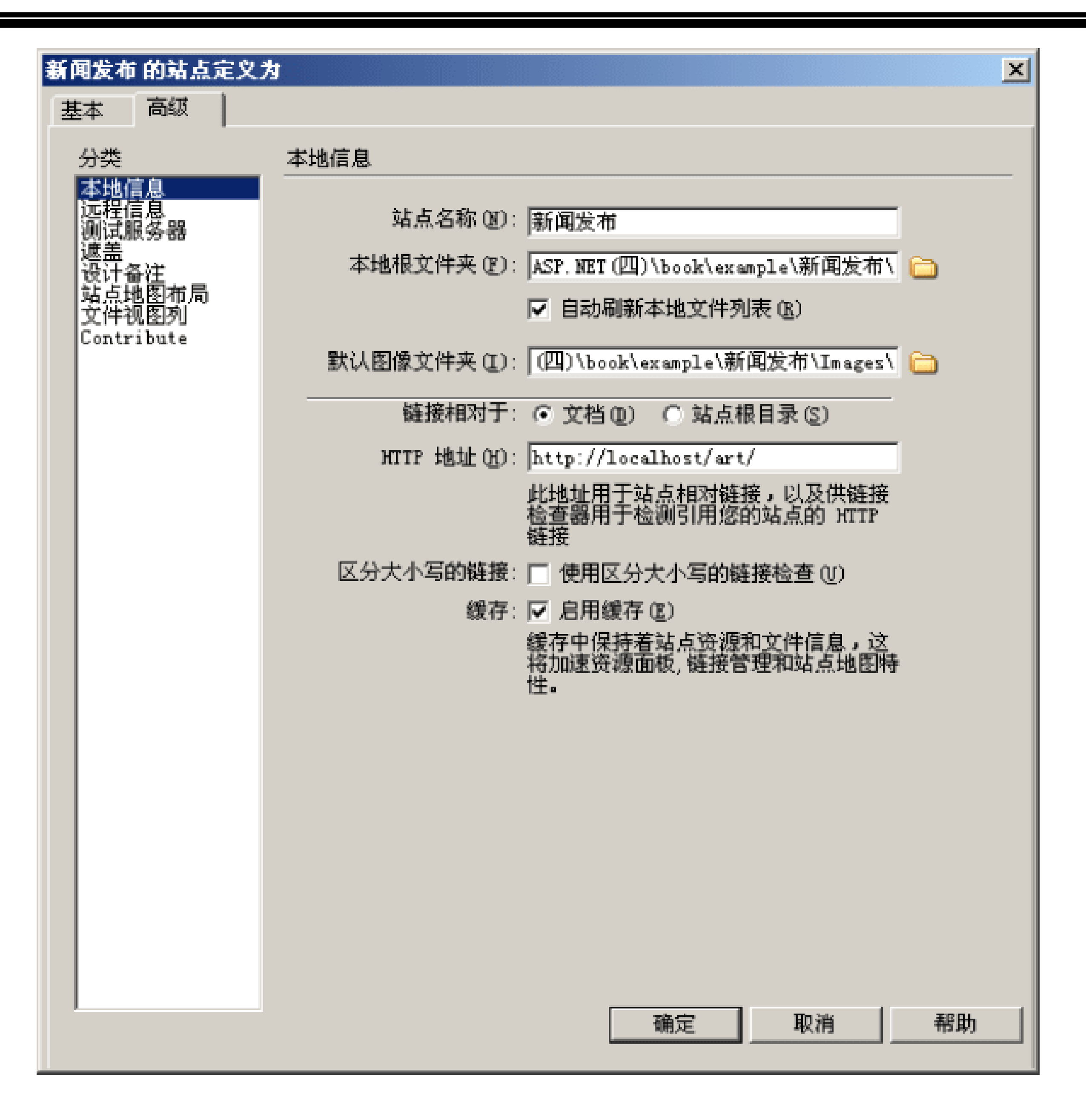


图 11.1　本地信息

其中，站点名称设置为“新闻发布”，本地根文件夹设置为新闻发布系统所在的磁盘目录，默认图像文件夹则设置为新闻发布系统下的 Images 目录，HTTP 地址设置为“http://localhost/art”(在此之前，需将新闻发布系统设置为 Web 共享，并设置其共享名为 art)。

站点设置中，远程信息设置如图 11.2 所示。

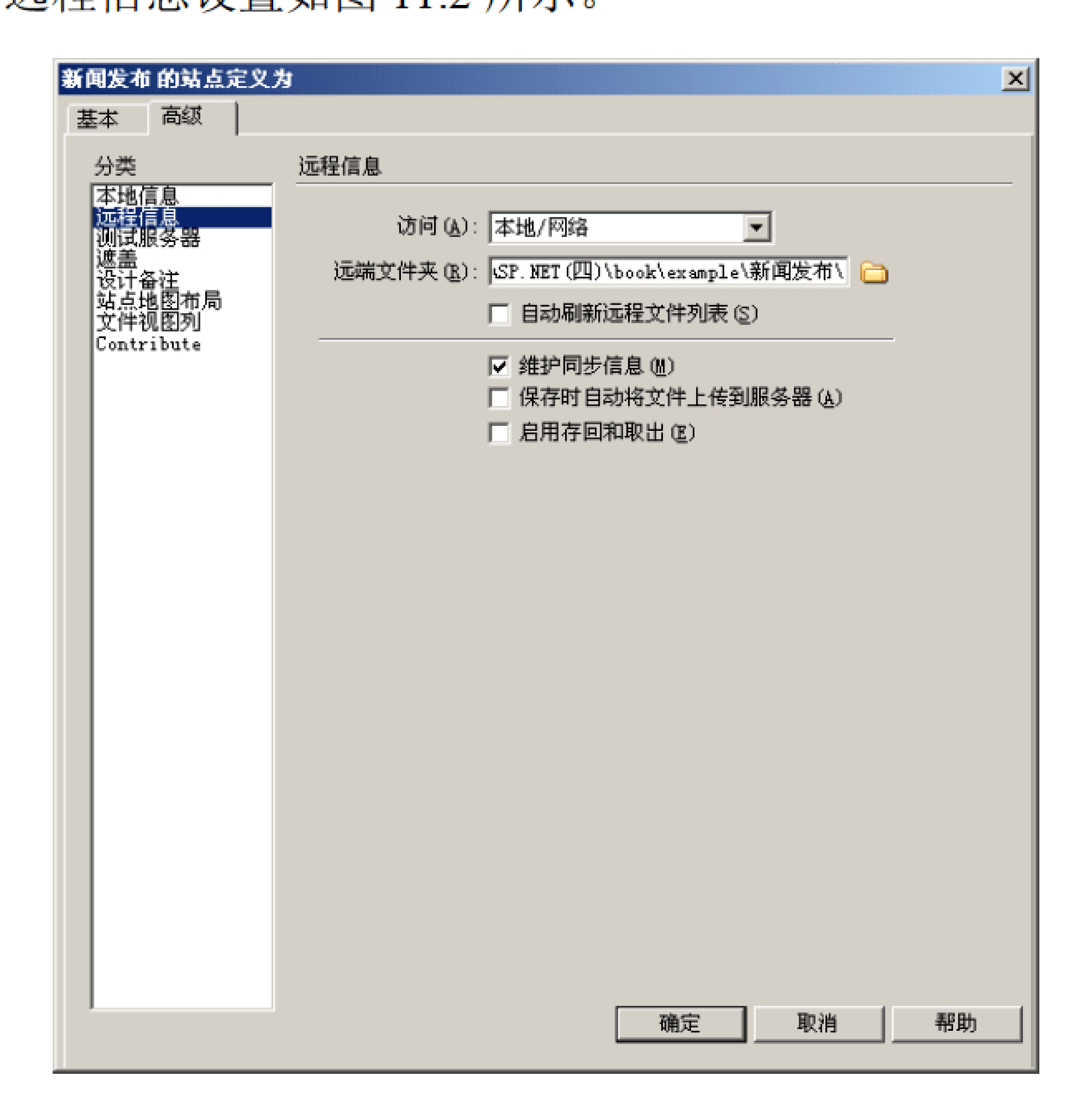


图 11.2　远程信息

由于对新闻发布系统的设计、测试和运行均是在本机上操作的，因此这里将访问方式

设置为【本地/网络】，而远端文件夹则与本地根文件夹设置为相同的目录。

站点设置中，测试服务器设置如图 11.3 所示。

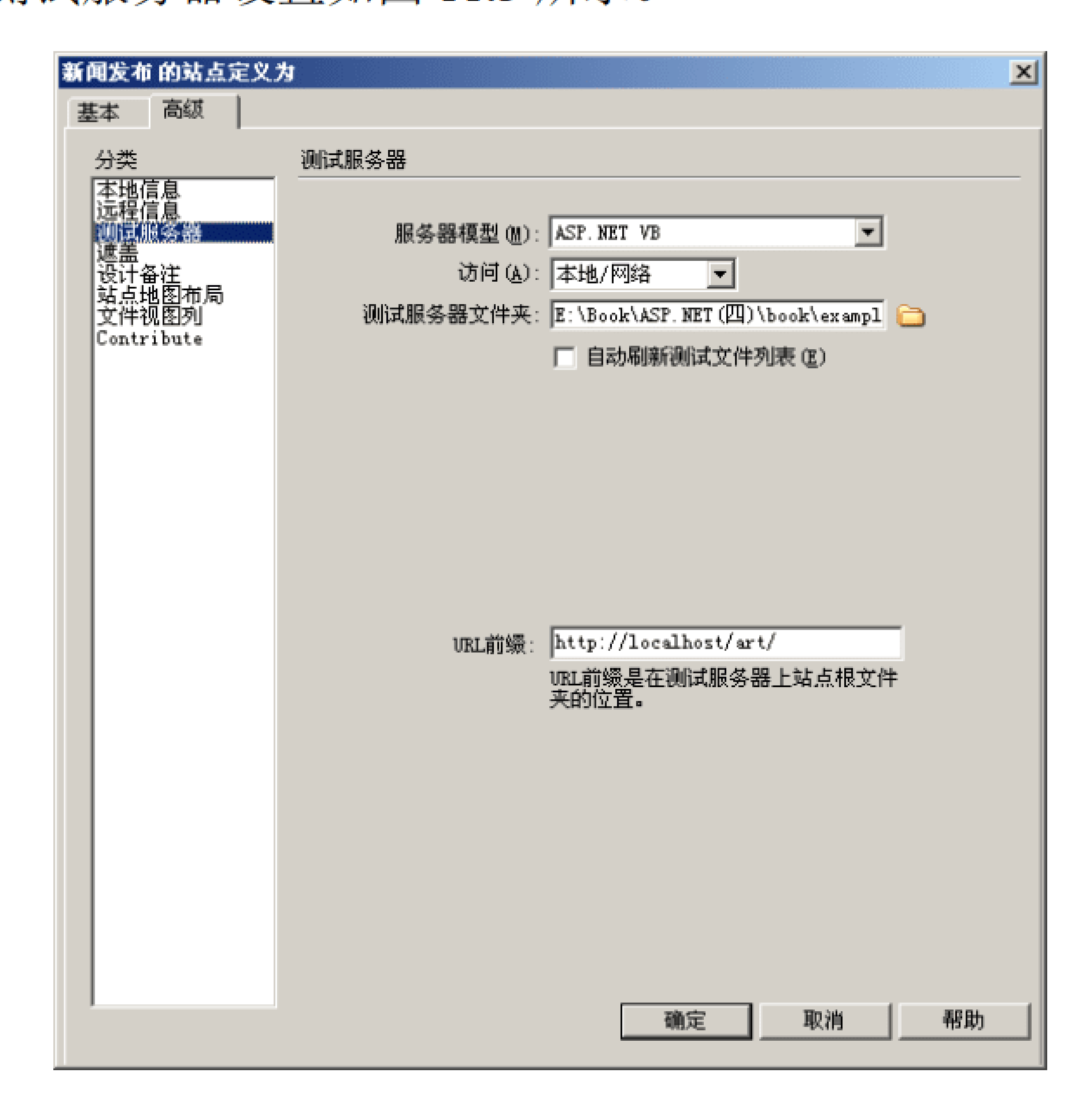


图 11.3　测试服务器

对于站点的部署，请参考本书第 7 章，这里不再赘述。

11.1.4　创建数据库连接

还需要创建一个数据库连接，这是通过 Dreamweaver 8 创建 ASP.NET 应用程序的一个不可少的操作。

新建一个 ASP.NET 页面，打开【应用程序】面板组，切换至【数据库】面板，单击 按钮，在弹出的菜单中选择【OLE DB 连接】命令，就会弹出【OLE DB 连接】对话框，设置连接名称为 Cnn，如图 11.4 所示。

图 11.4　创建数据库连接

单击【建立】按钮，在打开对话框的 Provider 选项卡中选择 Microsoft Jet 4.0 OLE DB Provider。然后切换至 Connection 选项卡，单击 ... 按钮，选择新闻发布系统目录下的 Data 子目录下的 Data.mdb 文件作为系统的数据库文件，如图 11.5 所示。

单击【确定】按钮，返回【OLE DB 连接】对话框，再次单击【确定】按钮，完成数据库的创建。

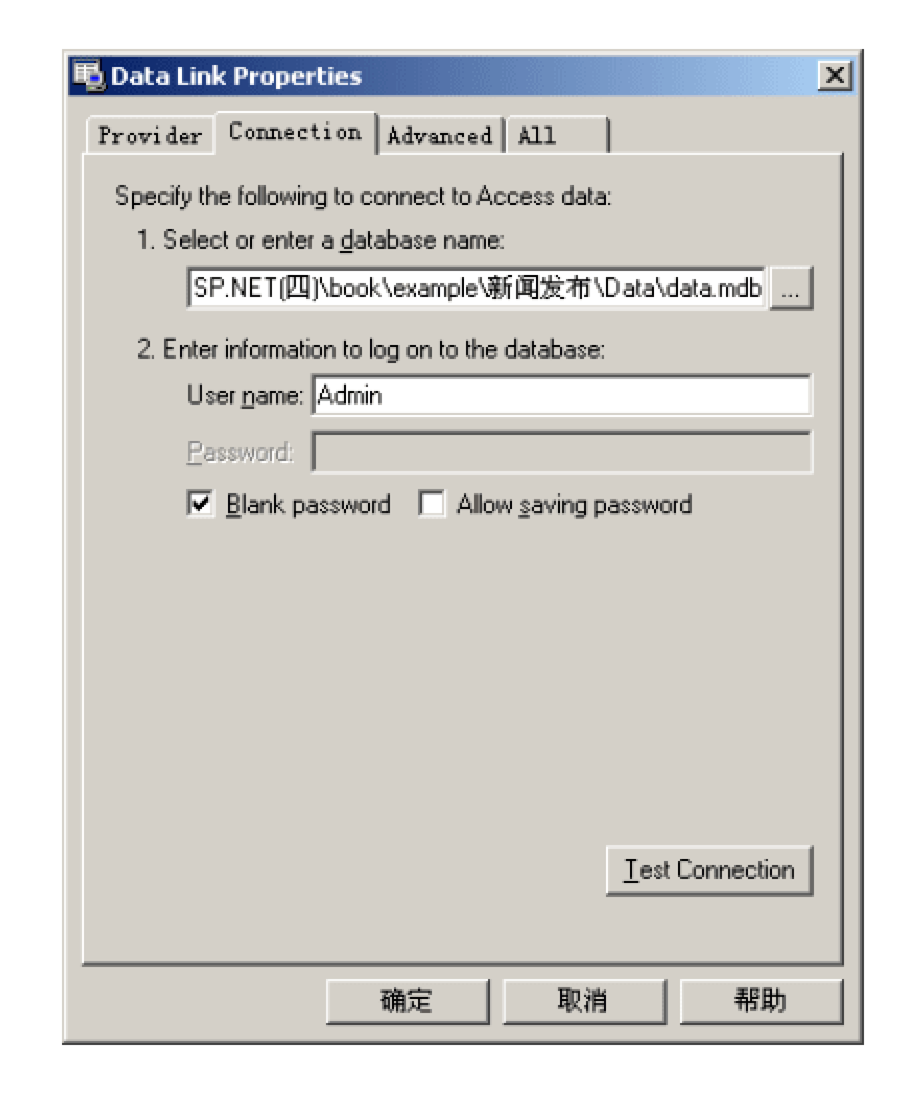


图 11.5　选择数据库

11.2　构 建 模 板

对于动态新闻发布系统来说，存在系统前台与后台之分，两者的页面布局也有所不同。因此，需要为前台页面和后台页面分别构建符合要求的模板。

11.2.1　前台模板

(1) 选择【文件】|【新建】命令，在弹出的【新建文档】对话框中，选择【模板页】|【ASP.NET VB 模板】，如图 11.6 所示。

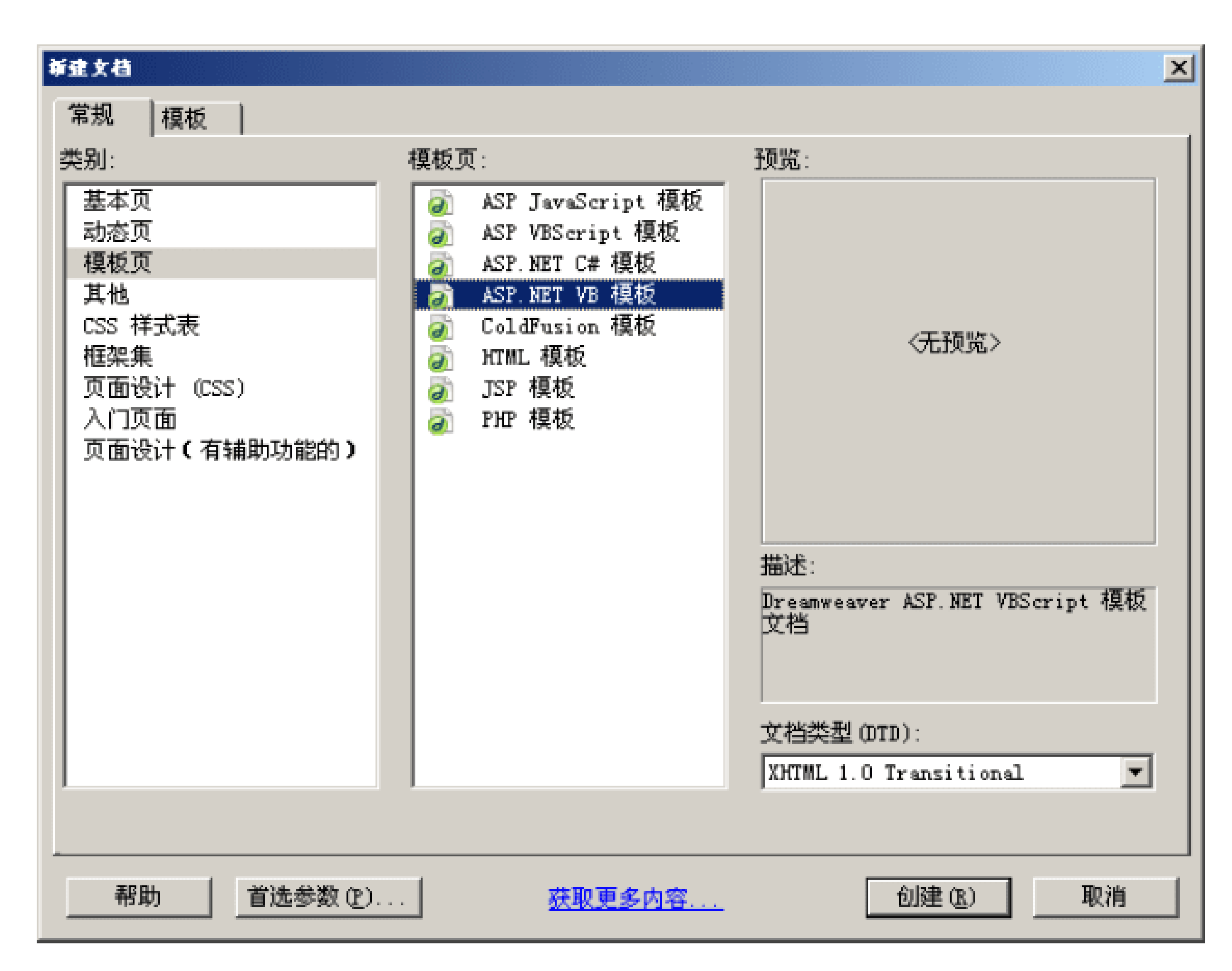


图 11.6　【新建文档】对话框

单击【创建】按钮，即可新建一个模板页，将文件保存为“前台模板”。

(2) 导入外部样式表。打开 CSS 面板，在该面板任意一处右击，从弹出的快捷菜单中选择【附加样式表】命令，此时将弹出【链接外部样式表】对话框，如图 11.7 所示。

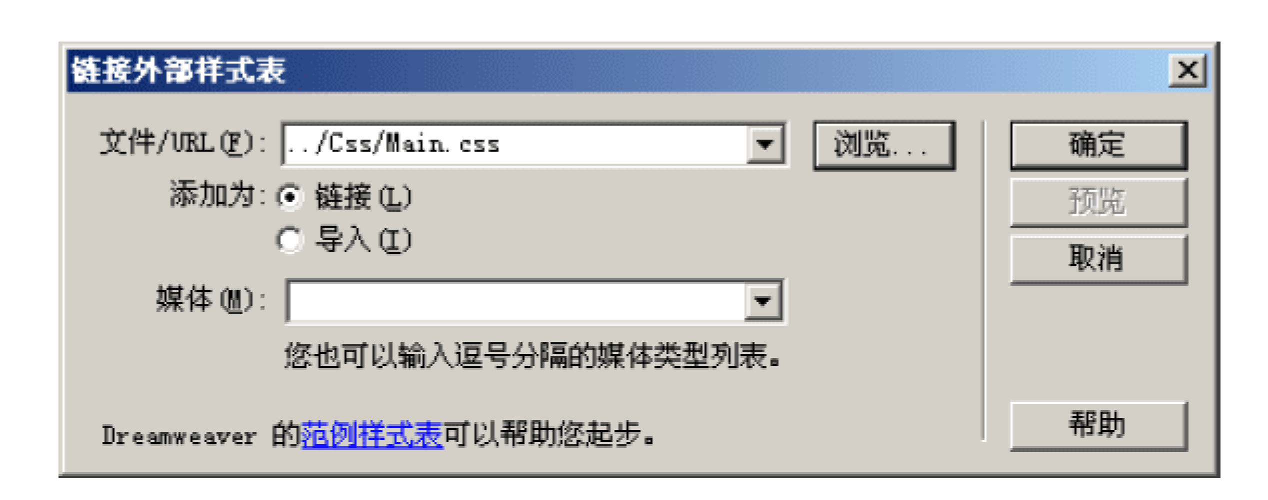


图 11.7　【链接外部样式表】对话框

(3) 单击【浏览】按钮，选择系统目录中的 CSS 文件夹下的 Main.css 文件，设置添加方式为【链接】。单击【确定】按钮，完成样式表的附加。

(4) 切换至【设计】视图，单击【属性】面板中【页面属性】按钮，在弹出的【页面属性】对话框中，设置页面的背景颜色值为“#d2d2ca”，上边距为 0。

(5) 选择【插入】|【表格】命令，在页面上插入一个 4 行 2 列的表格 table1，其属性设置如图 11.8 所示。

图 11.8　表格属性

(6) 合并表格第 1 行的两个单元格，并在其中插入一个图片，其源文件为 Images 目录下的 Top.gif。在第 2 行的两个单元格中，分别插入文本“今天日期：”和超级链接“管理”，并在第 1 个单元格的文本之后添加动态文本“<%=now().toString("yyyy 年 MM 月 dd 日")%>”，用于显示当前日期的年月日。

(7) 合并表格 table1 第 3 行的两个单元格，选择【插入】|【模板对象】|【可编辑区域】命令。在弹出的【新建可编辑区域】对话框，设置名称为 EditRegion1，如图 11.9 所示。

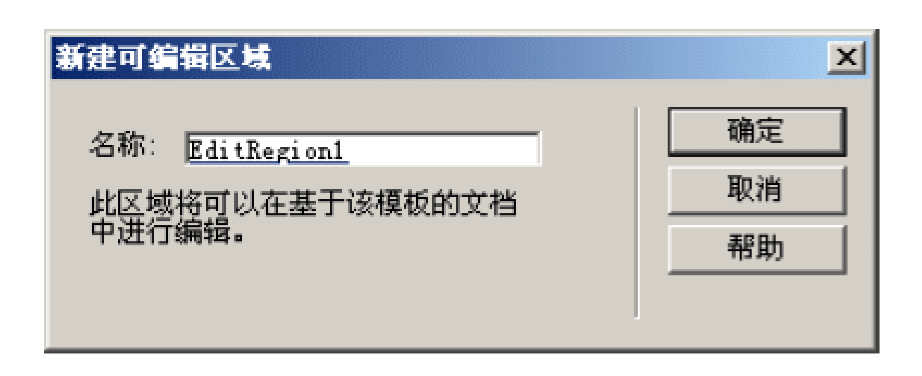


图 11.9　【新建可编辑区域】对话框

单击【确定】按钮，完成可编辑区域的创建。

(8) 在可编辑区域中，插入一个表单 Form，在【代码】视图中设置其 Runat 属性为 Server。同时添加 Style 样式，设置其属性 Margin-bottom 和 Margin-top 的值均为 0，如图 11.10 所示。

```
<!-- TemplateBeginEditable name="EditRegion1" -->
  <form runat="server" style="margin-bottom:0; margin-top:0">

  </form>
```

图 11.10　添加表单 Form

(9)　在表单 Form 中插入一个 1 行 3 列的表格 table2，其表格属性设置如图 11.11 所示。

图 11.11　表格属性

(10) 该表格的第 1 列用于放置新闻的分类信息和热门新闻链接，第 2 列用于实现左右两个单元格的分隔线，而第 3 列则用于放置页面的主体内部。

(11) 在表格 table2 的第 1 列中，再次插入一个 4 行 1 列的表格 table3，其表格设置如图 11.12 所示。

图 11.12　表格设计

(12) 在新闻分类下的单元格中插入了一个 ASP.NET 2.0 中的 TreeView 控件，由于 Dreamweaver 8 对其并不支持可视化操作，因此这里仅显示了一个ASP标记。TreeView 控件的定义代码如图 11.13 所示。

```
        <td height="300" align="left" valign="top" style="padding-left:15px;">
          <asp:TreeView  ID="TreeView1" runat="server" BorderStyle="None" ForeColor=
"Black" NodeWrap="True" ShowLines="True" ExpandDepth="1" />
        </td>
```

图 11.13　定义 TreeView 控件的定义代码

对于 TreeView 控件的数据绑定显示，是通过代码来执行的。切换至【代码】视图，添加页面的 Page_Load 事件和自定义过程 LoadClass()，代码如下：

【示例代码】

```
<Script Runat="Server">
  'Page_Load 事件
  Sub Page_Load(Sender As Object,E As EventArgs)
    If Not IsPostBack Then
```

```
      '调用自定义过程 LoadClass
      LoadClass()
    End If
  End Sub
  '自定义过程 LoadClass，用于获取数据库中的新闻分类信息，并将其绑定至 TreeView 控件
  Sub LoadClass()
    Dim Cnn As OleDbConnection
    Dim Cmd As OleDbCommand
    Dim DataR As OleDbDataReader
    Dim strConn As String
    Dim Sql As String
    Dim i, ParentNode, NodeId As Integer
    '获取数据库连接字符串
    strConn
=System.Configuration.ConfigurationSettings.AppSettings("MM_CONNECTION_S
TRING_Cnn")
    Cnn = New OleDbConnection(strConn)
    Cnn.Open()
    '查询新闻分类信息
    Sql = "select * from classinfo order by classid"
    Cmd = New OleDbCommand(Sql, Cnn)
    DataR = Cmd.ExecuteReader
    '定义节点数组，其下标对应于分类信息的 ID
    Dim menuNodes(100) As TreeNode
    For i = 0 To menuNodes.Length - 1
      '初始化节点
      menuNodes(i) = New TreeNode
    Next
    '循环记录集，依次加载 TreeView 节点
    Do While DataR.Read
      '获取上级分类信息对应的 ID
      ParentNode = DataR("upclass")
      '获取当前分类 ID
      NodeId = DataR("classid")
      '设置节点的显示文本
      menuNodes(NodeId).Text = DataR("classname")
      '设置节点的链接
      menuNodes(NodeId).NavigateUrl = "default.aspx?ClassID=" & NodeId
      If ParentNode <> 0 Then
        '当存在上级分类时，将本节点作为上级分类节点的子节点进行添加
        menuNodes(ParentNode).ChildNodes.Add(menuNodes(NodeId))
      Else
        '当不存在上级分类时，将本节点作为根节点进行添加
        TreeView1.Nodes.Add(menuNodes(NodeId))
      End If
    Loop
    Cnn.Close()  '关闭数据库连接
  End Sub
</Script>
```

对于热门新闻的数据显示，由于涉及数据集的创建，因此将在具体的功能页面中进行

数据的绑定。

至此，前台模板创建完成，其页面预览如图 11.14 所示。

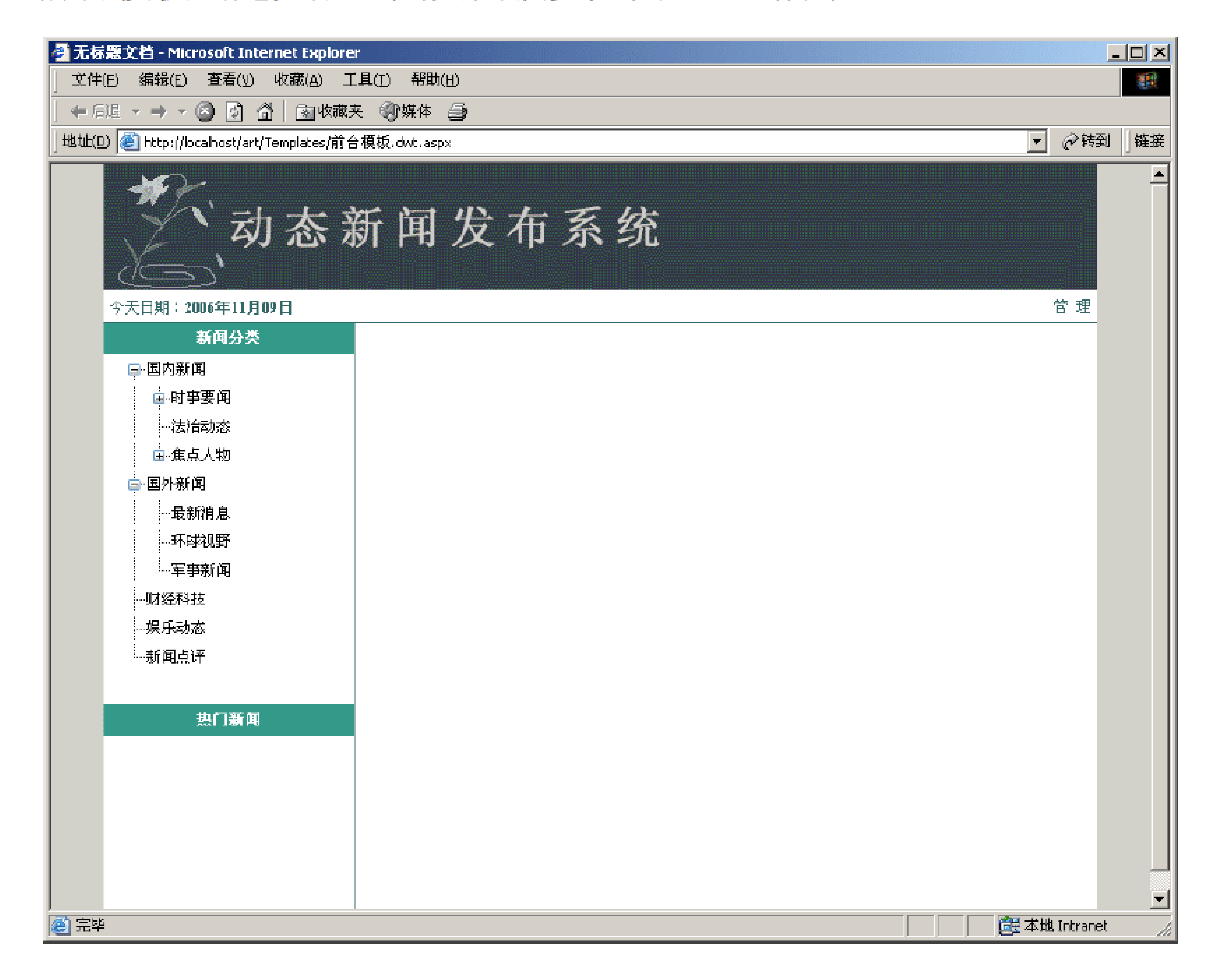


图 11.14　前台模板页面预览

这里，值得一提的是，虽然把相应的事件代码添加在页面模板中(如 Page_Load 事件)，但当将模板应用到具体的页面时，这些代码仍处于可编辑状态，用户仍可进行编辑，从而不会影响到具体页面的功能实现。

11.2.2　后台模板

后台模板的页面结构与前台模板的页面结构较为类似，其主要区别在于后台模板下方显示的并不是新闻分类和热门新闻，而是后台的功能模块链接。考虑两者的结构类似，因此无须重新创建模板，直接将前台模板另存，并在其基础上直接进行修改即可。

打开前台模板页面，选择【文件】|【另存为】命令，在弹出的【另存为】对话框中，将其另存为“后台模板.dwt.aspx”。

首先，打开【代码】视图，将页面的 Page_Load 事件及自定义过程 LoadClass 的代码删除；然后，切换至【设计】视图，将文本“管理”修改为“返回前台”，并更改其链接为“../default.aspx”。

调整表格 table3 的布局，其调整后的布局如图 11.15 所示。

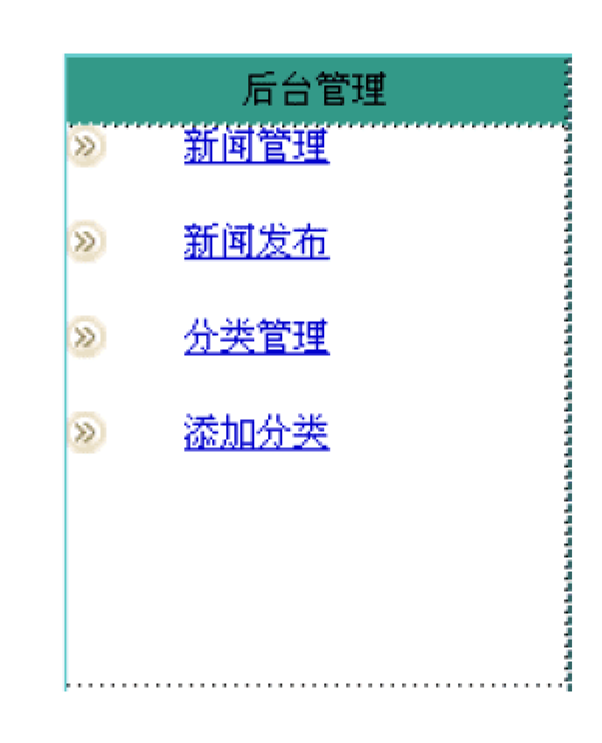


图 11.15　表格设计

其中，将文字“新闻管理”的链接设置为 Art_admin.aspx，“新闻发布”的链接设置为 Art_add.aspx，“分类管理”的链接设置为 Class_admin.aspx，“添加分类”

的链接设置为 Class_add.aspx，其目标属性均设置为_self。

至此，后台模板创建完成，其页面预览如图 11.16 所示。

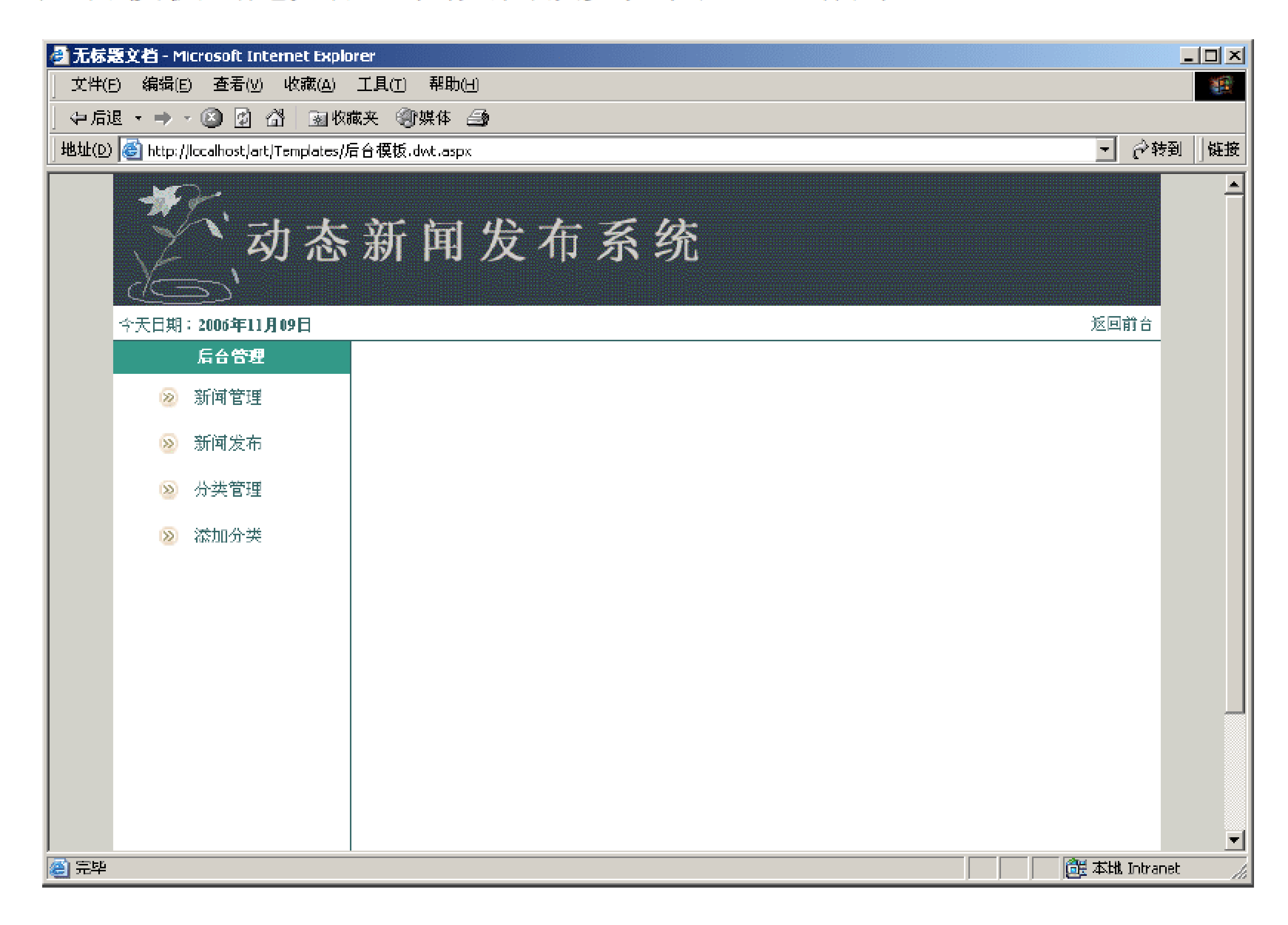


图 11.16　页面预览

11.3　新 闻 浏 览

在新闻浏览页面中，包括新闻列表和新闻搜索两个模块。前者，将显示指定分类下的所有新闻信息；后者，则提供给浏览者用来输入或选择新闻分类、新闻标题、关键字、新闻等内容的文本框或下拉列表框，以便通过多个条件搜索指定的新闻。

11.3.1　新闻列表

(1)　新建一个 ASP.NET VB 类型的页面，将其命名为 Default.aspx。选择【修改】|【模板】|【套用模板到页】命令，在弹出的【选择模板】对话框中选择【前台模板】，将模板应用到本页面。

先看一下热门新闻的数据绑定。

(2)　在【服务器行为】面板中单击按钮，从弹出的菜单中选择【数据集】命令，在弹出的【数据集】对话框中，设置数据集名称为 DataSet1，连接选择 Cnn，表格选择 ArtInfo，在【列】下拉列表框中选择字段 ID 和 ArtTitle，并设置排序方式为按字段 ClickNum 降序排序，如图 11.17 所示。

单击【确定】按钮，完成数据集 DataSet1 的创建。

(3)　在文字“热门新闻”下面的单元格中，插入一个<div>标签，并添加 Style 样式，设置其 Padding-bottom 属性的值为 5px；在<div>标签中插入一个图片，其源文件为 Images 目录下的 Arrow.gif。

(4)　打开【绑定】面板，选择并展开数据集 DataSet1，拖动字段 ArtTitle 至前面所插入

的图片之后，并在【属性】面板中设置其链接属性为“art_show.aspx? id=<%# DataSet1.FieldValue("id", Container) %>”。此时，单条数据的绑定已完成。

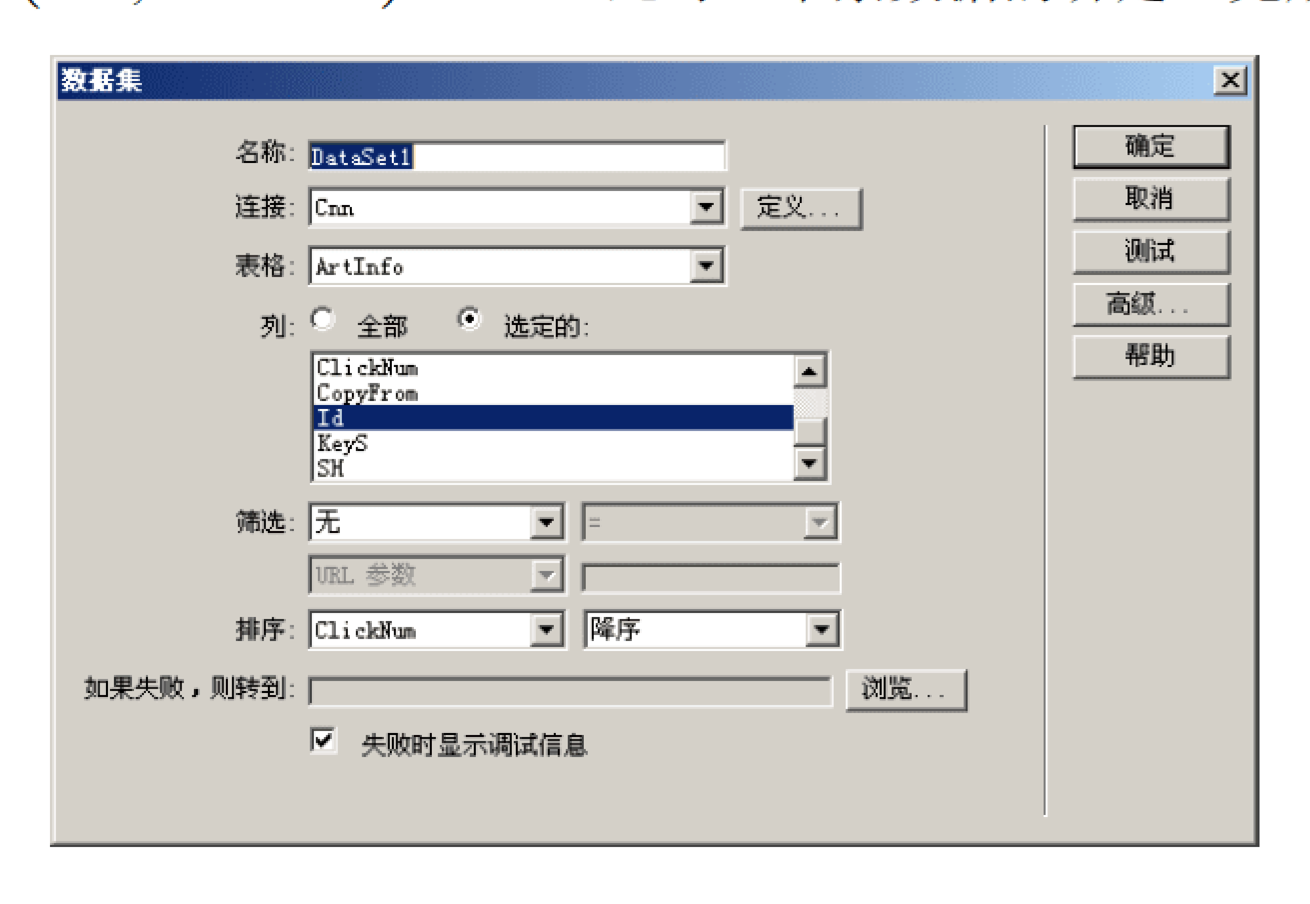


图 11.17　创建数据集 DataSet1

(5) 选择整个<div>标签，在【服务器行为】面板中单击按钮，从弹出的菜单中选择【重复区域】命令。在弹出的【重复区域】对话框中，选择数据集 DataSet1，设置一次显示 4 条记录，如图 11.18 所示。

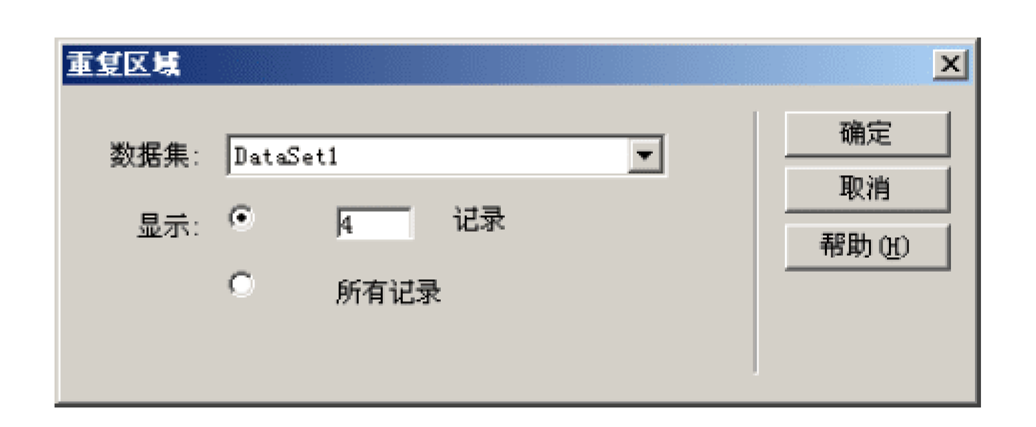


图 11.18　创建重复区域

(6) 单击【确定】按钮，完成重复区域的创建。在热门新闻中，仅显示 4 条点击率最高的新闻。为此，可以在数据集中设置仅返回 4 条记录(使用 Top 关键字)，或在重复区域中设置一次显示的记录数。这里，采用第二种方法。至此，热门新闻的数据绑定完成。

接下来，看看新闻列表功能的实现。

由于在新闻浏览页面中，包括了新闻列表和新闻搜索两个模块。因此，必须通过 Panel 控件来控制不同情况下的界面显示。

(7) 将光标置于表格 table2 的第 3 列的单元格中，在【ASP.NET-插入】工具栏中单击【更多标签】按钮，在弹出的【标签选择器】对话框中，选择【ASP.NET 标签】|【Web 服务器控件】分类，并在右侧服务器控件列表中选择 asp:Panel，如图 11.19 所示。

(8) 单击【插入】按钮，在弹出的【标签编辑器】对话框中设置其 ID 为 Panel1。单击【关闭】按钮，完成 Panel 控件的添加。

在新闻列表页面中，需要考虑两种情况：一种是用户直接进入本页面，而没有选择新闻分类，此时需要显示所有类别的新闻信息；另一种情况是当用户选择了某一新闻分类，仅显示该分类下的所有新闻。那么，如何判断用户是否选择了新闻分类呢？事实上，根据页面参数 ClassID 的值是否为空可以判断，此操作将在页面的 Page_Load 事件中执行。但必须添加相应的 Panel 控件，以针对不同的情况显示不同的信息。

图 11.19 【标签选择器】对话框

为此，在控件 Panel1 中，再次添加两个 Panel 控件，其 ID 分别设置为 Panel2 和 Panel3。

在显示新闻列表之前，需要创建两个数据集，分别针对以上两种不同情况的数据绑定。

(9) 在【服务器行为】面板中，单击 按钮，在弹出的菜单中选择【数据集】命令。单击【高级】按钮，切换至【数据集】对话框的高级模式。在【数据集】对话框的高级模式中，单击【编辑参数】按钮 ，在弹出的【编辑参数】对话框中设置参数名称为“@ClassId”，如图 11.20 所示。

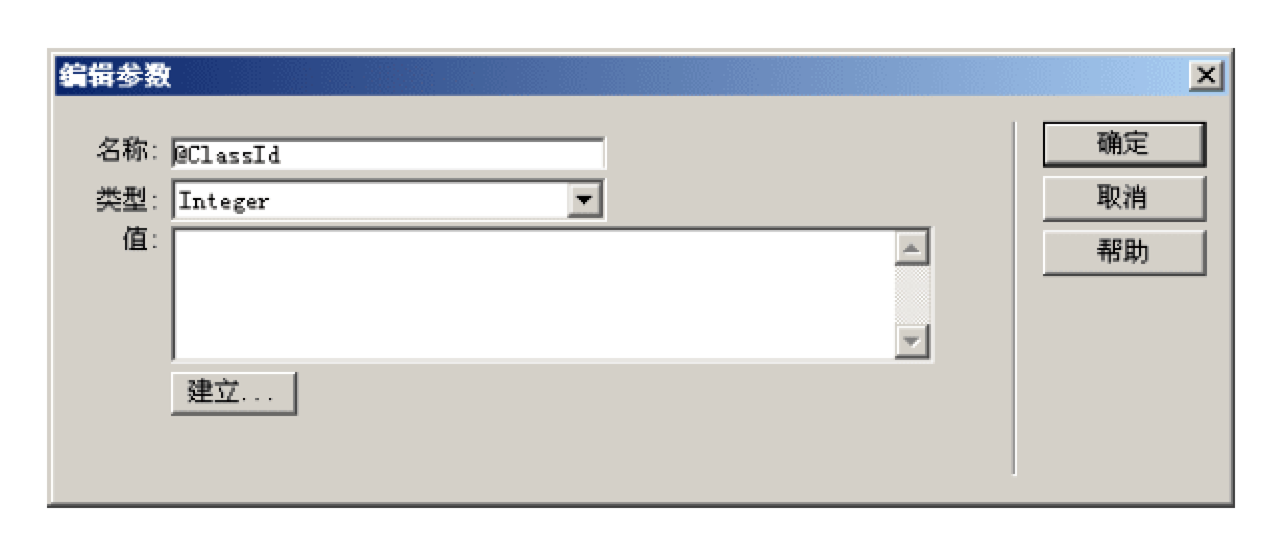


图 11.20 【编辑参数】对话框

(10) 单击【建立】按钮，在弹出的【生成值】对话框中，设置参数的值为 URL 参数 ClassId，如图 11.21 所示。

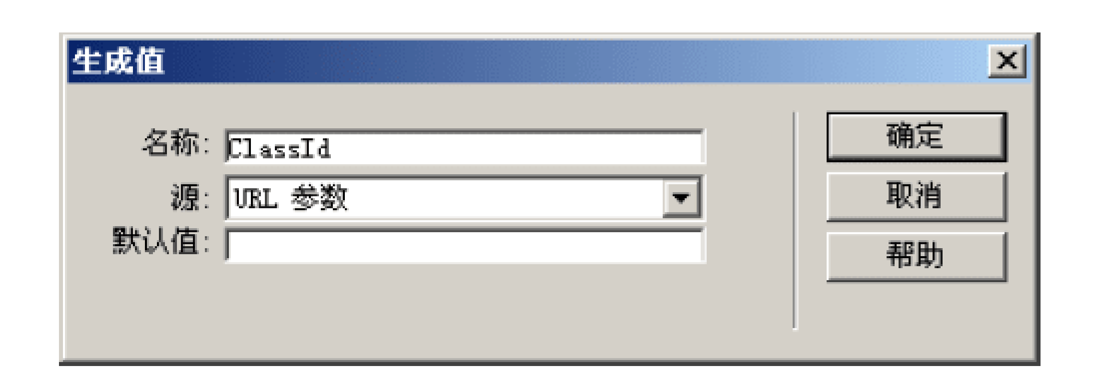


图 11.21 【生成值】对话框

(11) 单击【确定】按钮，返回【添加参数】对话框；再次单击【确定】按钮，返回【数据集】对话框。设置数据集的名称为 DataSet2，连接选择 Cnn，并在 SQL 文本框中输入以下 SQL 语句：

```
SELECT id,Addtime, ArtTitle, Author, CopyFrom,classname FROM
ArtInfo,ClassInfo WHERE ArtInfo.classid=ClassInfo.Classid and sh=1 and
ArtInfo.ClassId = ? ORDER BY addtime desc
```

数据集设置界面如图 11.22 所示。

单击【确定】按钮，完成数据集的创建，该数据集用于获取指定分类下的所有新闻信息。

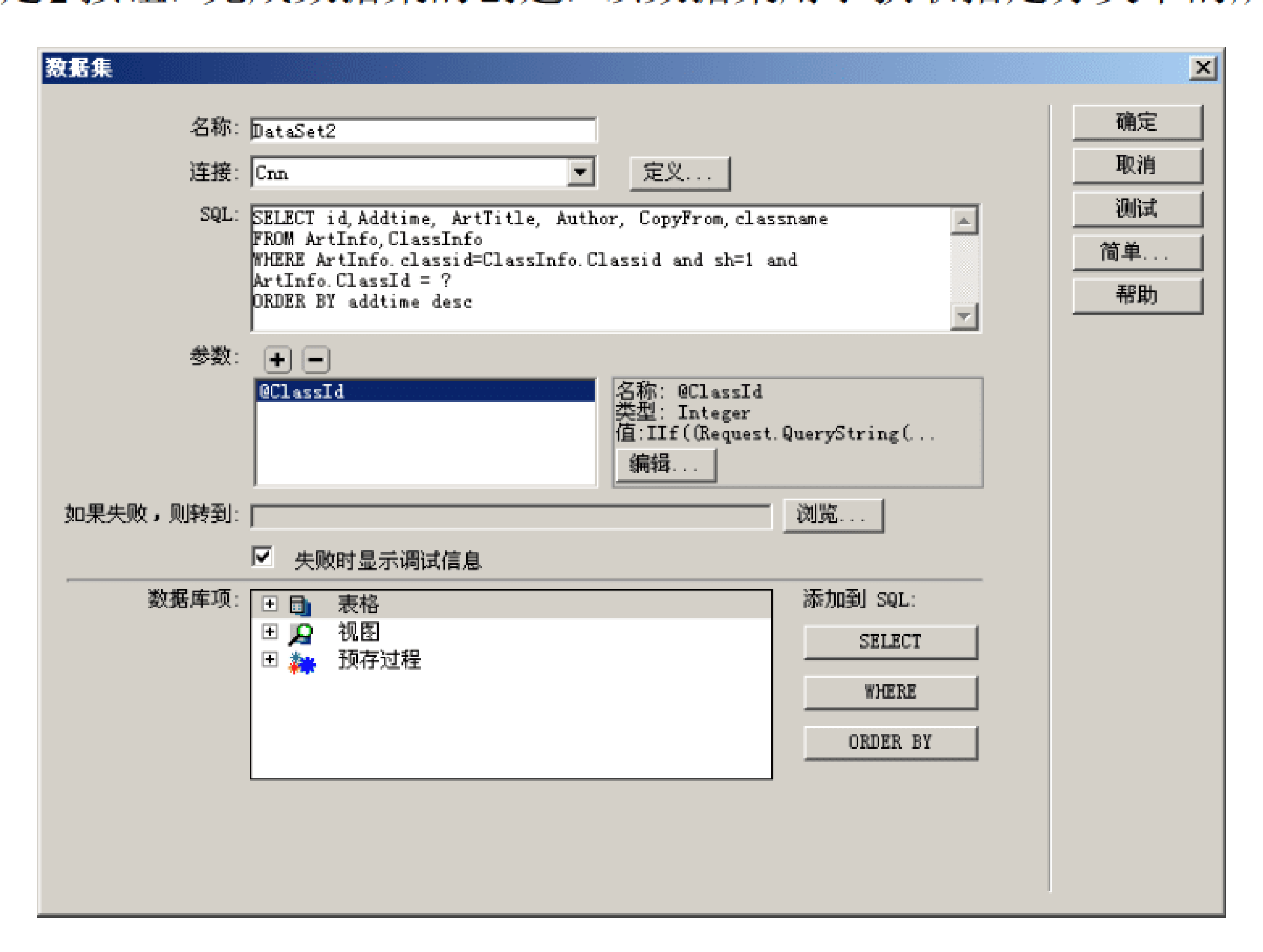


图 11.22　创建数据集 DataSet2

(12) 再次单击【服务器行为】面板中的+按钮，在弹出的菜单中选择【数据集】命令。单击【高级】按钮，切换至【数据集】对话框的高级模式。在【数据集】对话框的高级模式中，设置数据集的名称为 DataSet3，连接选择 Cnn，并在 SQL 文本框中输入以下 SQL 语句：

```
SELECT artinfo.id,arttitle,classname,author,copyfrom,addtime FROM
artinfo,classinfo WHERE artinfo.classid=classinfo.classid and sh=1 ORDER BY
addtime desc
```

如图 11.23 所示。

单击【确定】按钮，完成数据集 DataSet3 的创建，该数据集用于获取未指定分类情况下的所有新闻信息。

(13) 在控件 Panel2 中，输入文本“当前分类：”。在【应用程序】面板组中，切换至【绑定】面板，选择并展开数据集 DataSet2，拖动字段 ClassName 至该文本之后，并在【属性】面板中设置其字体颜色为红色，加粗显示。

(14) 按 Enter 键换行。在【服务器行为】面板中，单击+按钮，从弹出的菜单中选择【数据网格】命令。在弹出的【数据网格】对话框中，设置 ID 为“DataGrid1”，数据集选择 DataSet2，并设置为一次显示 10 条记录，【导航】选项设置为【编号链接到每一页】，如图 11.24 所示。

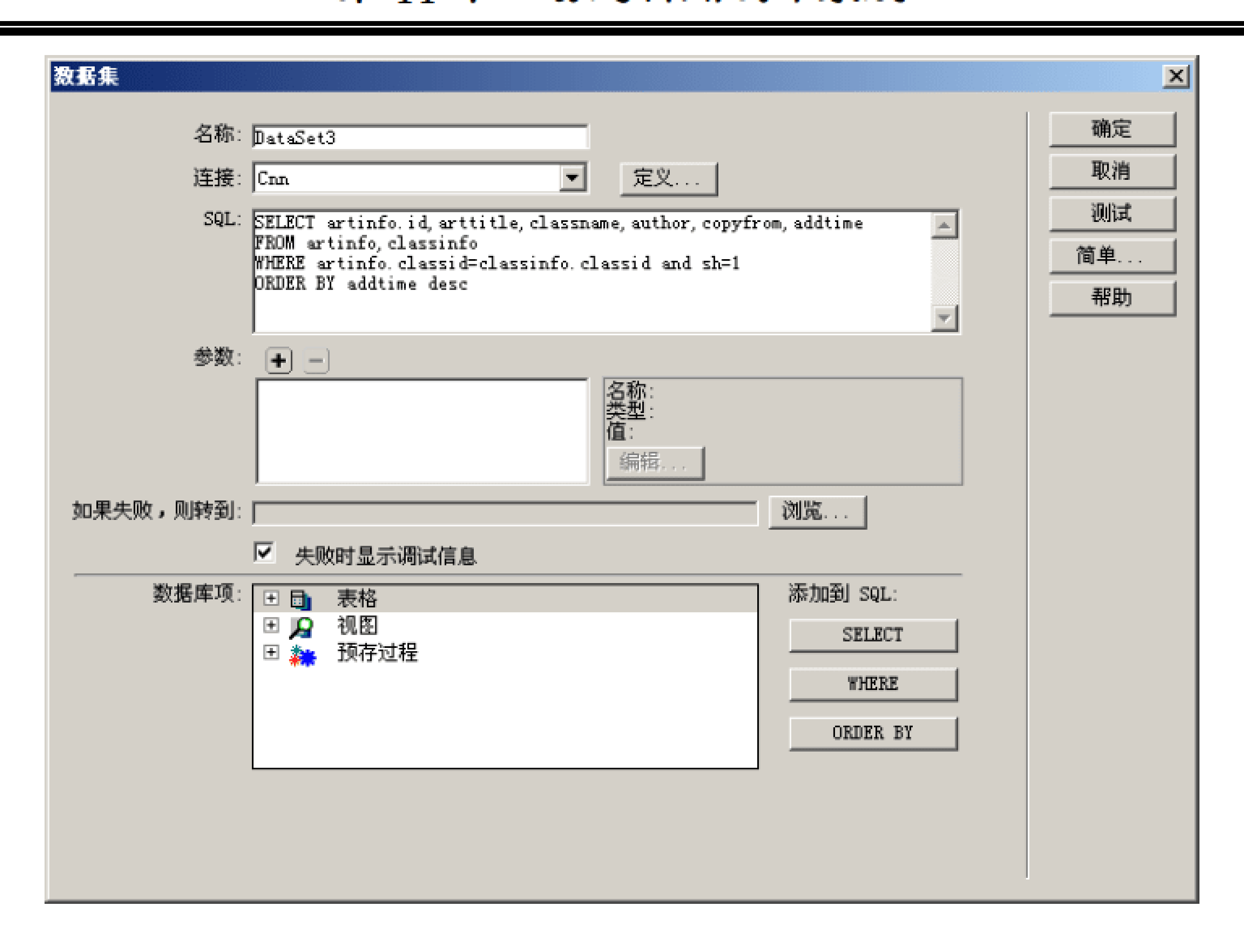


图 11.23　创建数据集 DataSet3

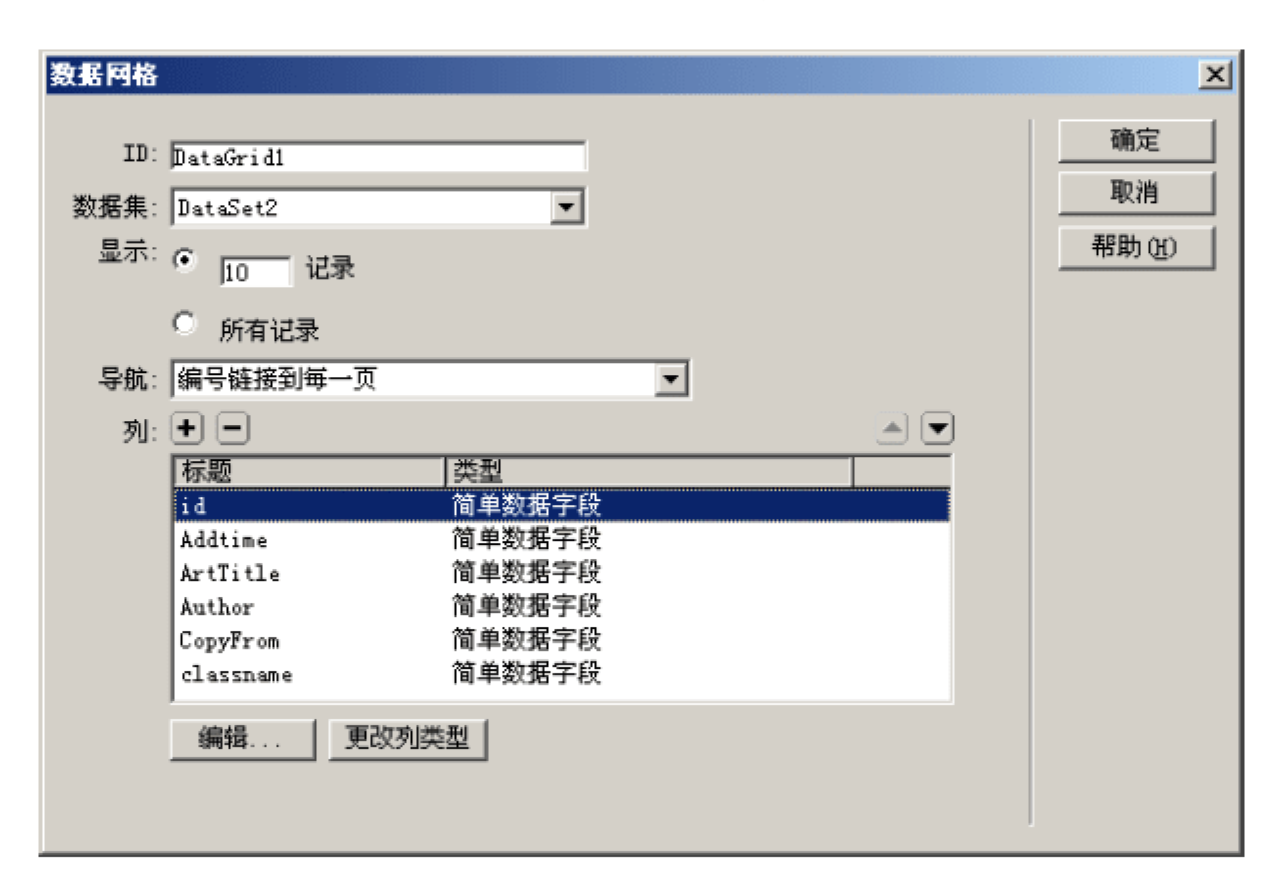


图 11.24　创建数据网格 DataGrid1

(15) 分别选择不需要在数据网格中显示的列 ID 和 ClassName，单击【删除列】按钮━，将其从【列】列表框中删除。选择 Author 列，单击【编辑】按钮，在弹出的【简单数据字段列】对话框中，设置其标题为“作者”，如图 11.25 所示。

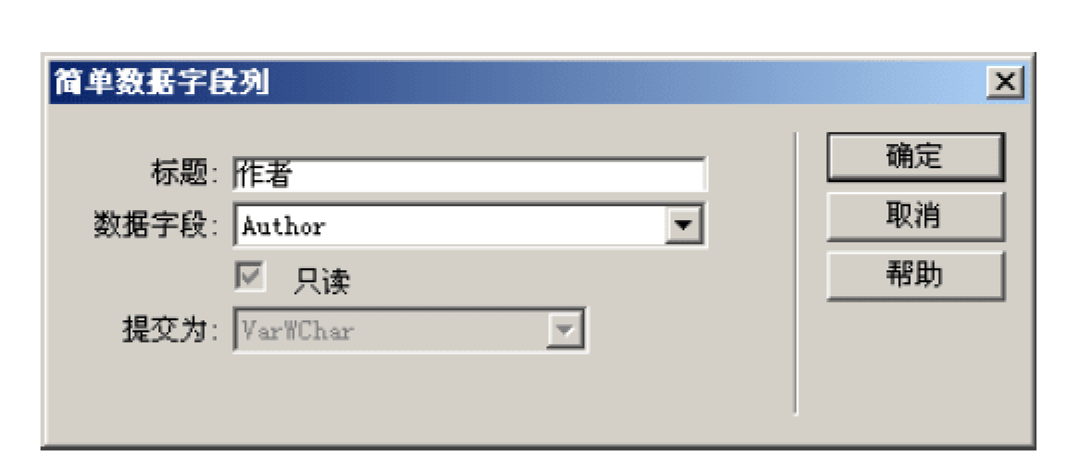


图 11.25　【简单数据字段列】对话框

(16) 重复此操作，分别设置列 Addtime 和 CopyFrom 的显示标题为“发布时间”和“新闻来源”。选择列 ArtTitle，单击【更改列类型】按钮，从弹出的菜单中选择【超级链接】命令，此时将弹出【超级链接列】对话框，如图 11.26 所示。

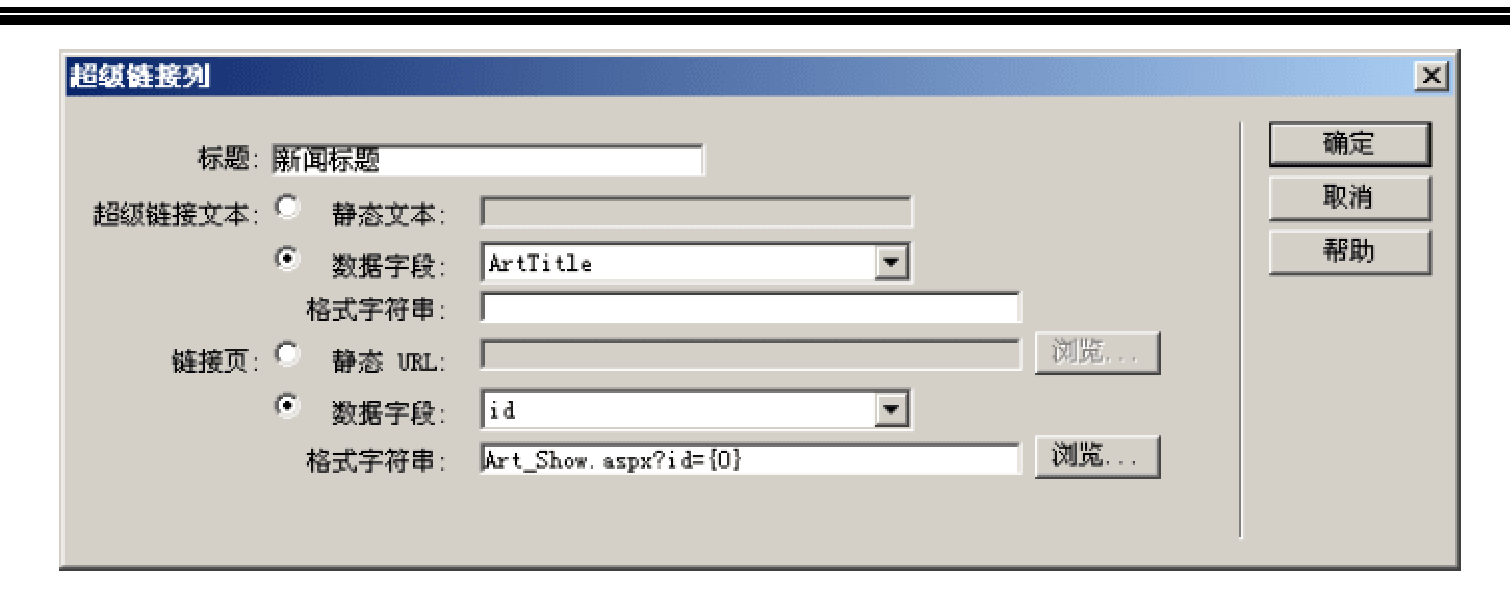


图 11.26　【超级链接列】对话框

(17) 设置【标题】为“新闻标题”，【超级链接文本】为数据字段 ArtTitle，【链接页】为数据字段 ID，【格式字符串】设置为“Art_Show?id={0}”。单击【确定】按钮，完成超级链接列的设置。

(18) 通过单击▲和▼按钮，调整字段列的顺序从上到下为“新闻标题”、“作者”、“新闻来源”和“发布时间”，如图 11.27 所示。

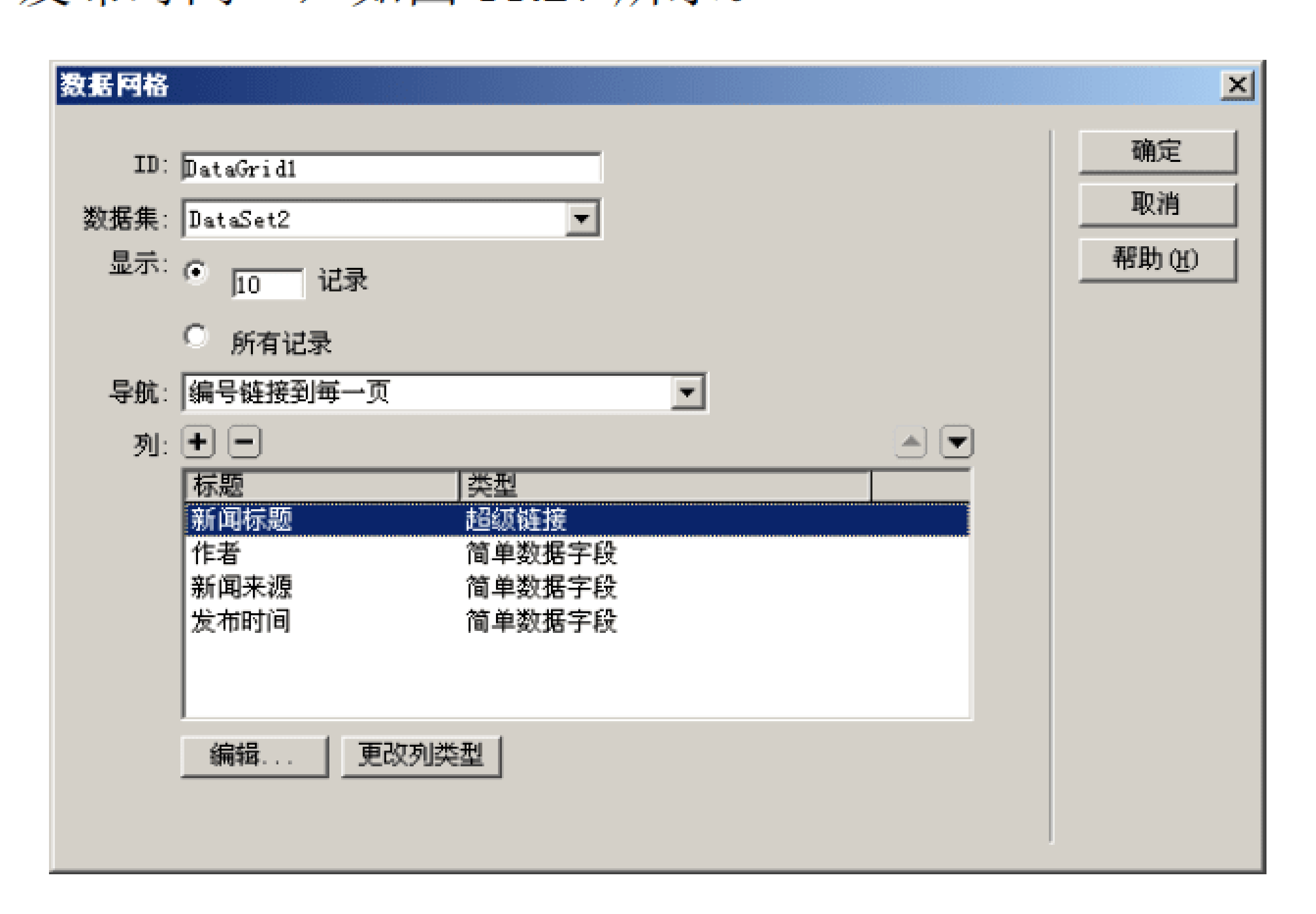


图 11.27　【数据网格】对话框

单击【确定】按钮，完成数据网格 DataGrid1 的创建。

(19) 为了控制数据集为空时的显示，在数据网格 DataGrid1 之后，输入文本“该分类下暂无相关新闻！”，并设置其字体颜色为红色。选择该段文本，在【服务器行为】面板中单击+按钮，在弹出的菜单中选择【显示区域】|【数据集为空时显示】命令，并在弹出的对话框中设置数据集为 DataSet2，如图 11.28 所示。

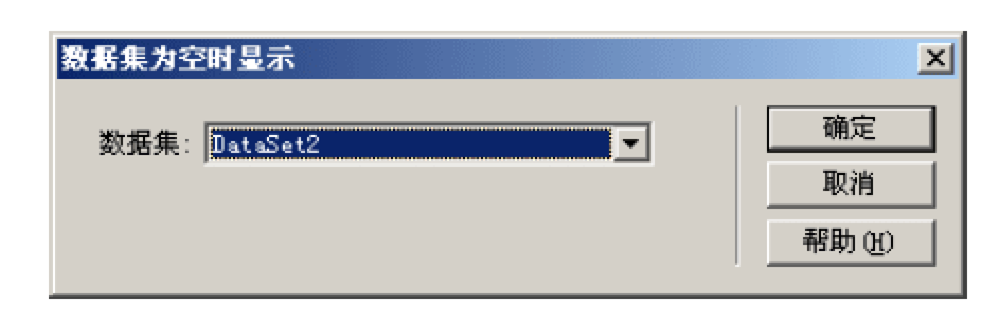


图 11.28　【数据集为空时显示】对话框

单击【确定】按钮，完成显示区域的设置。

(20) 选择数据网格 DataGrid1，在【服务器行为】面板中单击+按钮，在弹出的菜单中选择【显示区域】|【数据集不为空时显示】命令，并在弹出的对话框中同样设置数据集为

DataSet2。

(21) 将光标置于控件 Panel3 中，打开【服务器行为】面板，单击 按钮，在弹出的菜单中选择【数据网格】命令，在弹出的【数据网格】对话框中设置 ID 为 DataGrid2，数据集选择 DataSet3，设置为一次显示 10 条记录，在【导航】下拉列表框中选择【编号链接到每一页】，如图 11.29 所示。

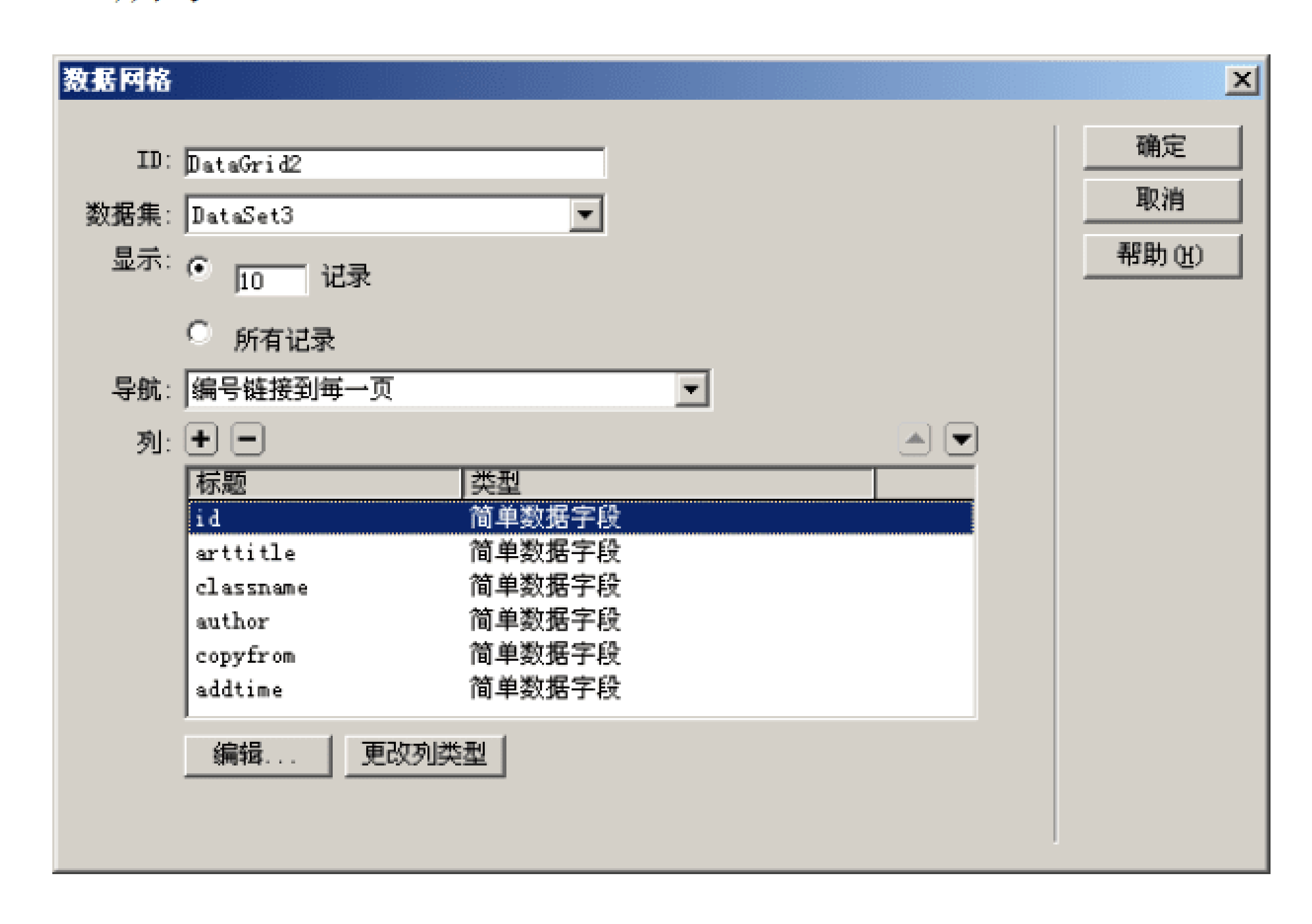


图 11.29　创建数据网格 DataGrid2

(22) 选择列 ID，单击【删除列】按钮 ，将其从【列】列表框中删除。选择 ClassName 列，单击【编辑】按钮，在弹出的【简单数据字段列】对话框中，设置其标题为“所属分类”。重复此操作，分别设置列 Author、Addtime 和 CopyFrom 的显示标题为“作者”、“发布时间”和“新闻来源”。选择列 ArtTitle，单击【更改列类型】按钮，从弹出的菜单中选择【超级链接】命令。在弹出的【超级链接列】对话框中设置【标题】为“新闻标题”，【超级链接文本】为数据字段 ArtTitle，【链接页】为数据字段 id，【格式字符串】设置为“Art_Show?id={0}”，如图 11.30 所示。

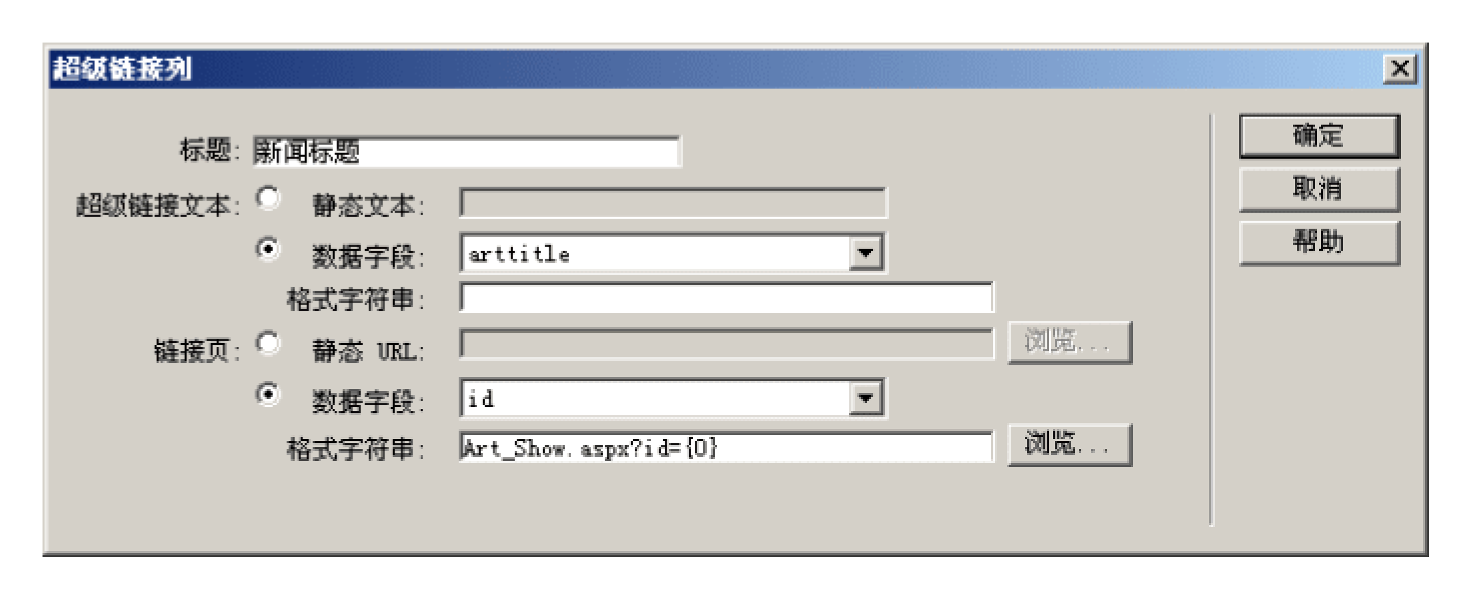


图 11.30　【超级链接列】对话框

(23) 单击【确定】按钮，完成超级链接列的设置。最后，通过单击 和 按钮，调整字段列的顺序从上到下为“新闻标题”、“所属分类”、“作者”、“新闻来源”和“发布时间”，如图 11.31 所示。

单击【确定】按钮，完成数据网格 DataGrid2 的创建。

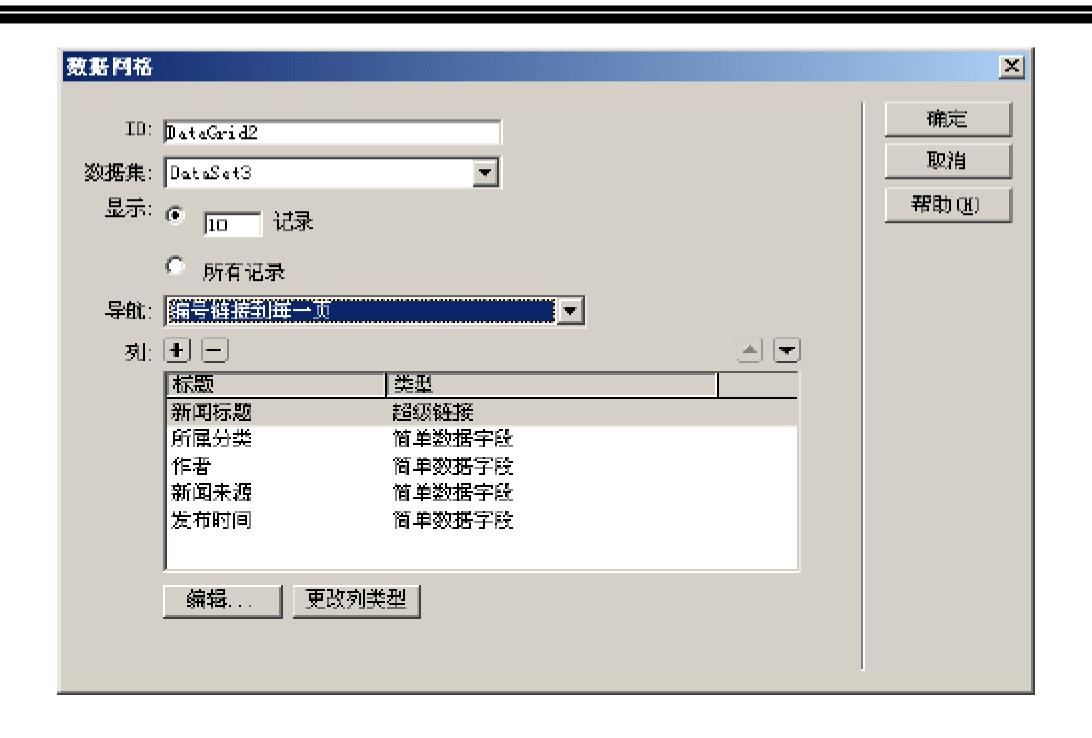


图 11.31　数据网格 DataGrid2

(24) 切换至【代码】视图，对页面的 Page_Load 事件进行修改，其代码如下：

【示例代码】

```
Sub Page_Load(Sender As Object,E As EventArgs)
  If Not IsPostBack Then
    LoadClass()
    '判断页面参数 ClassID 是否为空，以此判断用户是否选择了新闻分类
    if request.QueryString("ClassID")="" Then
      '页面参数 ClassID 为空，显示所有新闻信息
      Panel2.visible=false
      Panel3.visible=true
    else
      '页面参数 ClassID 不为空，显示指定分类下的所有新闻信息
      Panel2.visible=true
      Panel3.visible=false
    end if
  End If
End Sub
```

至此，新闻浏览页面中新闻列表的功能已经全部完成。

当用户初次打开新闻发布系统的主页面时，将显示所有分类的新闻信息，其页面预览效果如图 11.32 所示。

图 11.32　页面预览效果

当用户选择某一新闻分类时，可查看此分类下的所有新闻，其页面预览效果如图 11.33 所示。

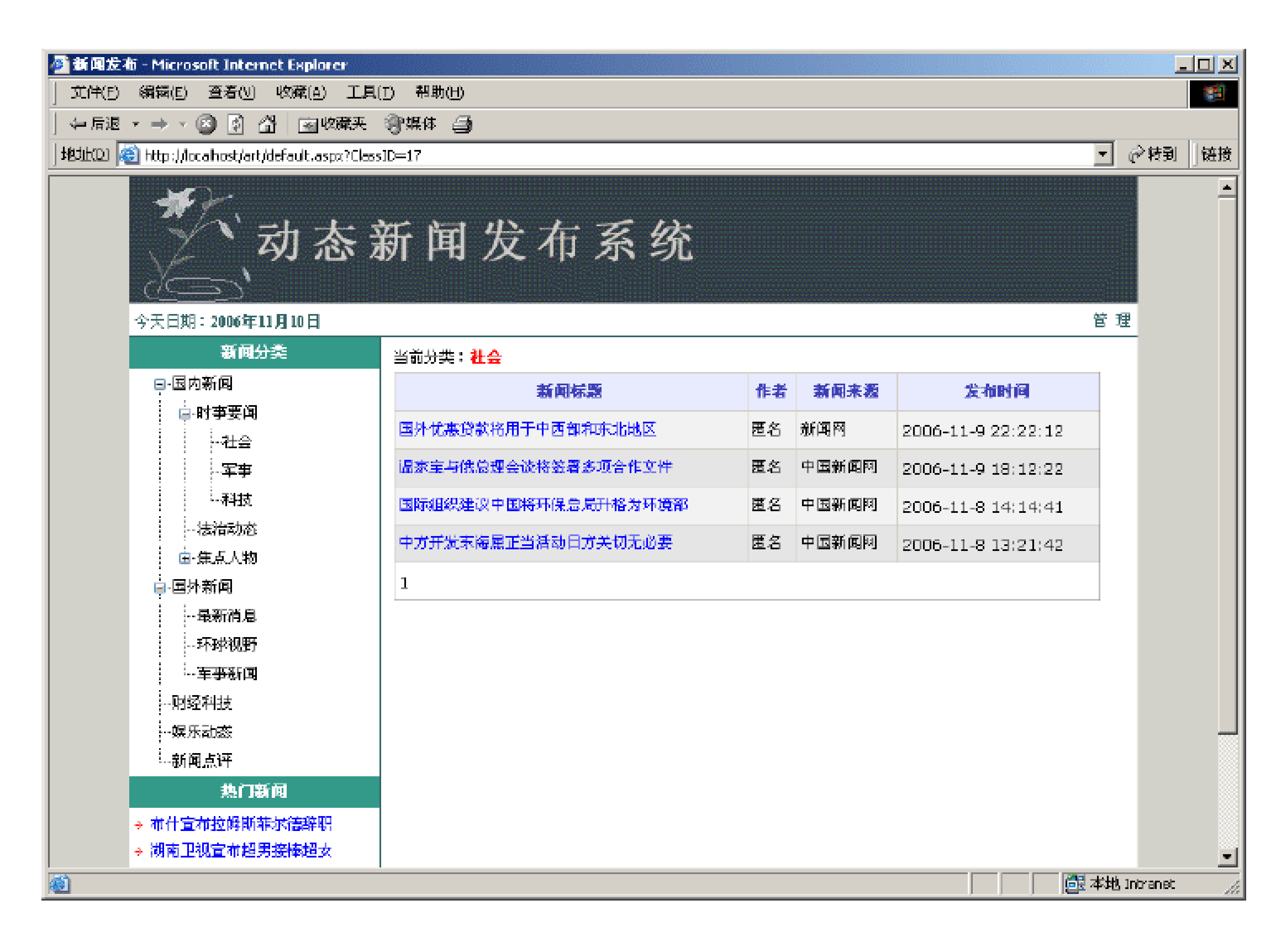


图 11.33　页面预览效果

11.3.2　新闻搜索

下面，再来看新闻浏览页面中新闻搜索功能的实现。

(1) 在控件 Panel1 之后，插入一个 2 行 1 列的表格 table4，其表格属性设置如图 11.34 所示。

图 11.34　表格属性

(2) 在表格的各个单元格中输入相应的文本及设置搜索条件用的文本框或下拉列表框控件，如图 11.35 所示。

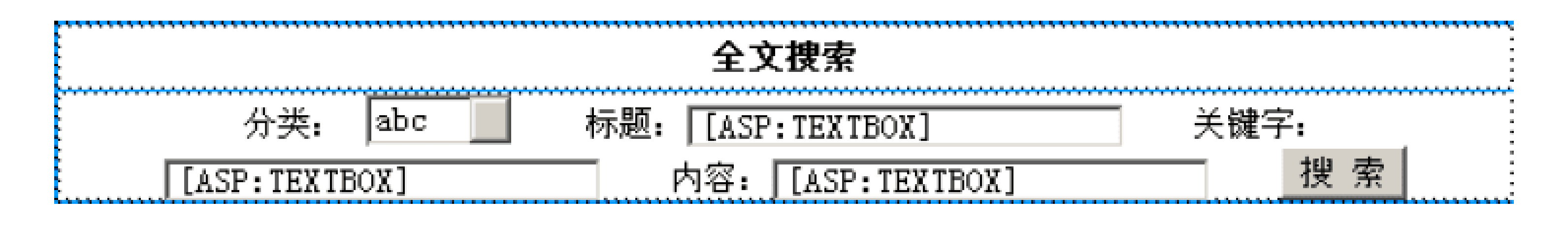


图 11.35　表格设计

(3) 设置三个文本框控件的 Width 属性均为 60px，以保证搜索条件能在一行显示。对于“分类”处的下拉列表框，其数据的绑定均将通过自定义过程 LoadClass1()来执行，其代码如下：

【示例代码】

```
Sub LoadClass1(ByVal ClassId As Integer, ByVal TheTag As Integer)
  '过程参数 ClassID 表示父分类的 ID，过程参数 TheTag 表示当前分类的级别
  Dim Cnn As OleDbConnection
```

```
  Dim Cmd As OleDbCommand
  Dim DataR As OleDbDataReader
  Dim Sql As String
  Dim Str1 As String
  Dim i as Integer
  Dim StrCnn As String
  '获取数据库连接字符串
  StrCnn =
System.Configuration.ConfigurationSettings.AppSettings("MM_CONNECTION_ST
RING_Cnn")
  Cnn = New OleDbConnection(StrCnn)
  Cnn.Open()  '连接数据库
  '查询当前父分类下的所有分类信息
  Sql = "select * from classinfo where upclass=" & trim(ClassID) & " order
by classid"
  Cmd = New OleDbCommand(Sql, Cnn)
  DataR = Cmd.ExecuteReader
  For i = 1 To TheTag
    '根据当前分类的级别，获取分类名称的前缀
    Str1 = Str1 & "├"
  Next
  While DataR.Read
    '定义 DropDownList 控件的数据项
    Dim theItem As New ListItem()
    '设置数据项的显示文本
    theItem.Text = trim(Str1) & trim(DataR("classname"))
    theItem.Value = DataR("classid")  '设置数据项的返回值
   '将当前定义的数据项添加至 DropDownList 控件中
    ssfl.Items.Add(theItem)
    '嵌套调用过程 LoadClass1，加载当前分类下的子分类
    LoadClass1(DataR("classid"), TheTag+1)
  End While
  Cnn.Close()  '关闭数据库连接
End Sub
```

为了获取搜索结果，需要再次创建一个数据集。

(4) 打开【应用程序】面板组，切换至【服务器行为】面板，单击 ± 按钮，在弹出的菜单中选择【数据集】命令。单击【高级】按钮，切换至【数据集】对话框的高级模式。在【数据集】对话框的高级模式中，设置数据集的名称为 DataSet4，连接选择 Cnn。

(5) 通过单击【添加参数】按钮 + ，依次添加 4 个参数：@ClassID、@Title、@Key 和@Content，其取值分别设置为表单变量 SSFL、Title、Key 和 Content 的值，即分别获取于 4 个搜索条件的输入值。

(6) 在 SQL 文本框中输入以下 SQL 语句：

```
SELECT artinfo.id,arttitle,classname,author,copyfrom,addtime FROM
artinfo,classinfo WHERE artinfo.classid=classinfo.classid and sh=1 and
artinfo.classid=? and arttitle like '%'+?+'%' and keys like '%'+?+'%' and
artcontent like '%'+?+'%' ORDER BY addtime desc
```

如图 11.36 所示。

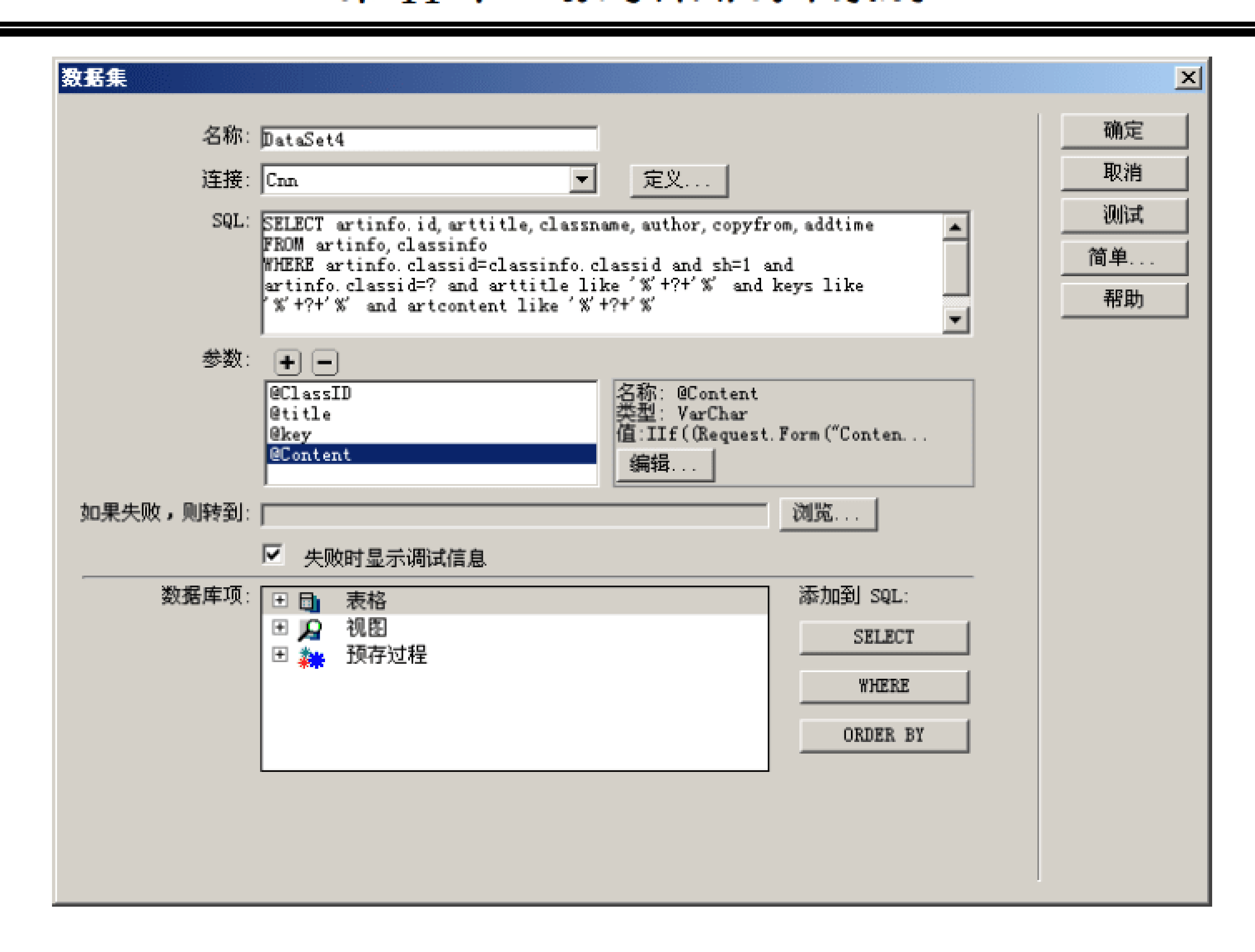


图 11.36　创建数据集 DataSet4

单击【确定】按钮，完成数据集的创建，该数据集将返回所有符合搜索条件的新闻信息。

(7)　在表格 table4 之后，插入一个 Panel 控件，设置其 ID 为 Panel4，该控件用于控制搜索结果区域的显示。

(8)　将光标置于 Panel4 中，在【服务器行为】面板中，单击 + 按钮，从弹出的菜单中选择【数据网格】命令。在弹出的【数据网格】对话框中，设置 ID 为 DataGrid3，数据集选择 DataSet4，并设置一次显示 10 条记录，在【导航】下拉列表框中选择【编号链接到每一页】，如图 11.37 所示。

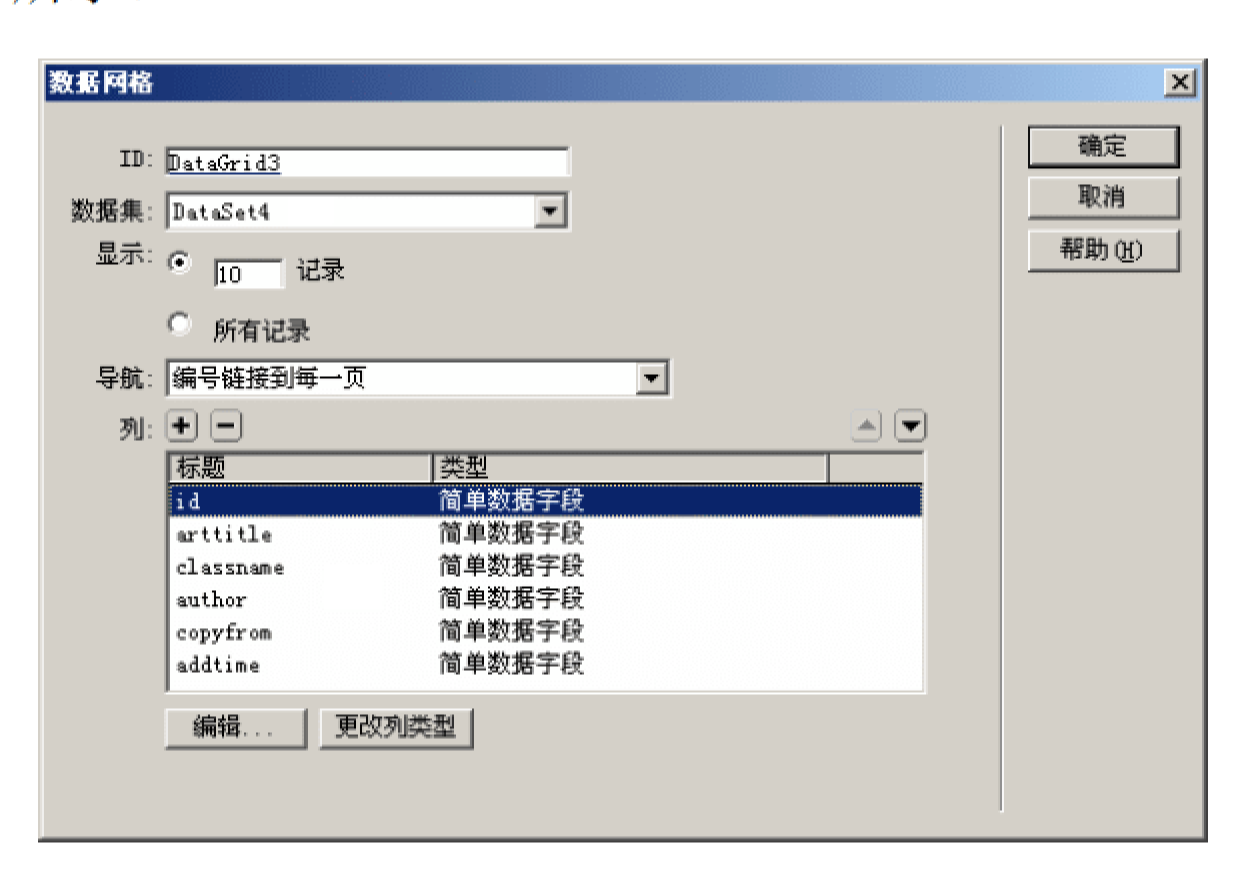


图 11.37　创建数据网格 DataGrid3

(9)　选择列 ID，单击【删除列】按钮 − ，将其从【列】列表框中删除。分别设置列 ClassName、Author、Addtime 和 CopyFrom 的显示标题为“所属分类”、“作者”、“发布时间”和“新闻来源”。选择列 ArtTitle，单击【更改列类型】按钮，从弹出的菜单中选择【超级链接】命令。在弹出的【超级链接列】对话框中，设置【标题】为“新闻标题”，【超级链接文本】为数据字段 arttitle，【链接页】为数据字段 id，【格式字符串】设置为

“Art_Show.aspx?id={0}”，如图 11.38 所示。

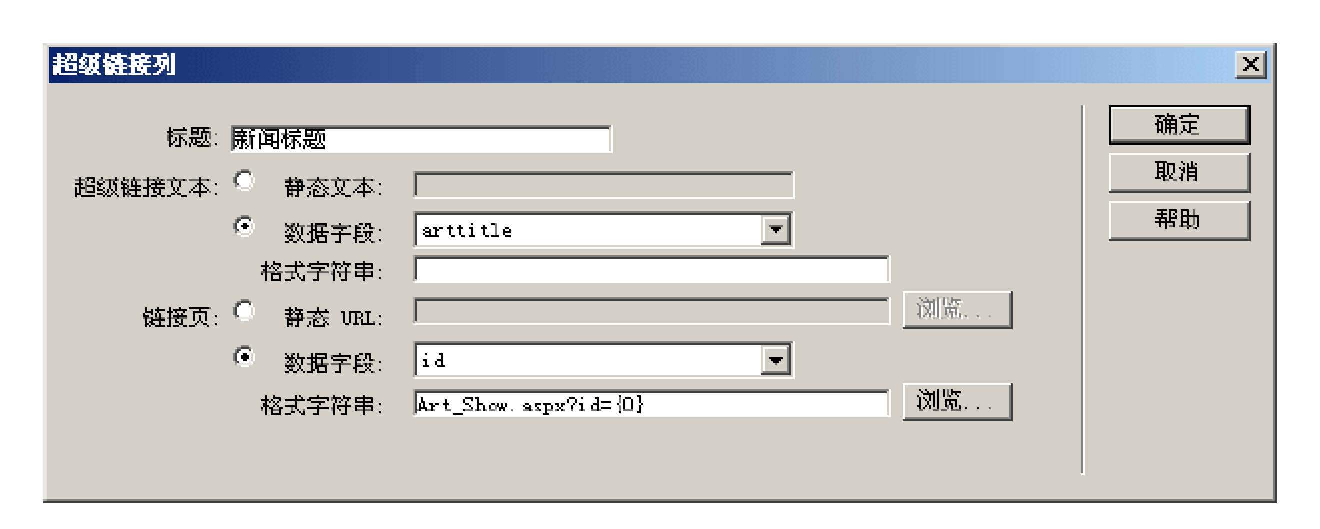


图 11.38　【超级链接列】对话框

(10) 单击【确定】按钮，完成超级链接列的设置。最后，通过单击和按钮，调整字段列的顺序从上到下为新闻标题、所属分类、作者、新闻来源和发布时间，如图 11.39 所示。

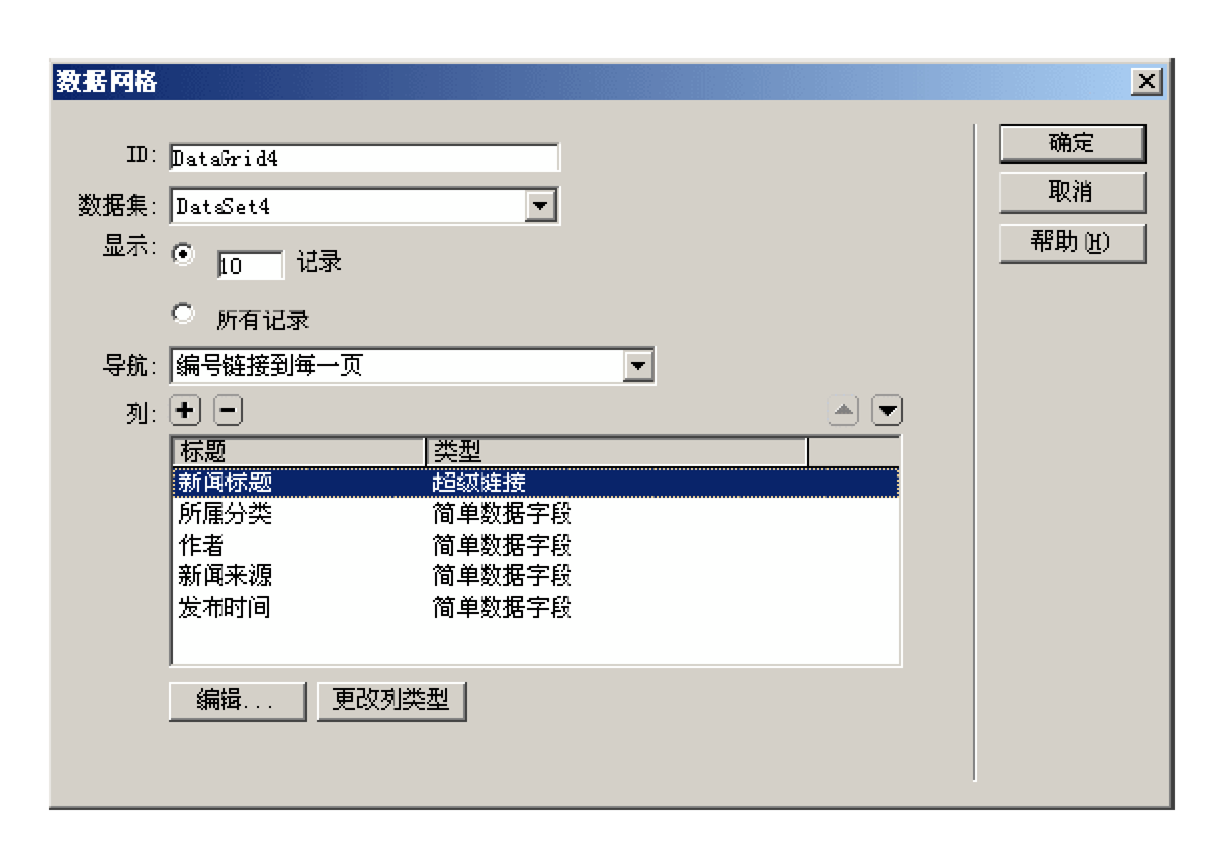


图 11.39　数据网格 DataGrid4

单击【确定】按钮，完成数据网格 DataGrid3 的创建。

(11) 在【设计】视图中，选择数据网格 DataGrid4，在【服务器行为】面板中单击按钮，在弹出的菜单中选择【显示区域】|【数据不为空时显示】命令。在弹出的【数据集不为空显示】对话框中，选择数据集 DataSet4，如图 11.40 所示。

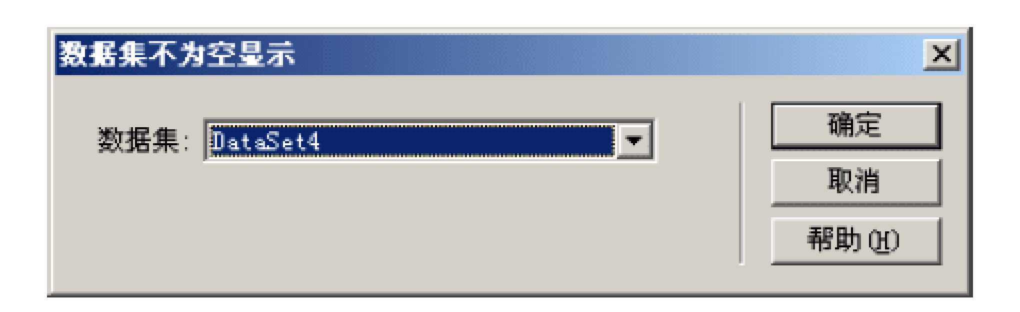


图 11.40　【数据集不为空显示】对话框

单击【确定】按钮，完成显示区域的设置。

(12) 在以上显示区域之后，输入文本“没有您想要的新闻，请重新搜索！”，并设置其字体颜色为红色。选择该文本，在【服务器行为】面板中，单击按钮，从弹出的菜单中选择【显示区域】|【数据为空时显示】命令，并在弹出的【数据为空时显示】对话框中选择数据集 DataSet4。

(13) 此时，新闻搜索的页面设计基本完成。最后，再来看 Page_Load 事件的调整，其

事件代码如下：

【示例代码】

```
Sub Page_Load(Sender As Object,E As EventArgs)
  If Not IsPostBack Then
    '加载页面左侧的新闻分类树形列表框
    LoadClass()
    '加载搜索条件中的新闻分类下拉列表框
    LoadClass1(0,0)
    '当页面初次加载时，显示新闻列表，隐藏搜索结果区域
    Panel1.visible=true
    panel4.visible=false
    if request.QueryString("ClassID")="" Then
      Panel2.visible=false
      Panel3.visible=true
    else
      Panel2.visible=true
      Panel3.visible=false
    end if
  else
    '当页面由数据回发导致提交时，显示搜索结果区域，隐藏新闻列表
    panel1.visible=false
    panel4.visible=true
  End If
End Sub
```

至此，新闻浏览页面的功能全部设计完成。

当用户查看指定分类或所有分类的新闻列表时，页面预览效果如图 11.41 所示。

图 11.41　页面预览效果

当输入搜索条件并单击【搜索】按钮后，页面将显示搜索结果并隐藏新闻列表，如图 11.42 所示。

图 11.42　搜索结果

11.4　新 闻 阅 读

新闻阅读是指查看指定的新闻信息。在新闻阅读页面中，除了显示当前的新闻信息，还需添加【上一条】链接和【下一条】链接，实现浏览同一新闻分类下的上一条新闻和下一条新闻。

(1)　新建一个 ASP.NET　VB 类型的页面，将其命名为 Art_Show.aspx。选择【修改】|【模板】|【套用模板到页】命令，在弹出的【选择模板】对话框中选择【前台模板】，将模板应用到本页面。

实现热门新闻的数据绑定。为此，需要创建数据集 DataSet1 和一个重复区域，此操作与新闻列表中的相关操作完全相同，这里不再赘述。

对于单条新闻的阅读，其指定的新闻 ID 可由页面变量 ID 来获取。在绑定新闻相关的动态数据之前，先来创建一个数据集。

(2)　在【服务器行为】面板中，单击按钮，在弹出的菜单中选择【数据集】命令。设置数据集名称为 DataSet2，选择连接 Cnn，表格设置为 ArtInfo，筛选条件为字段 Id 等于 URL 参数 Id，如图 11.43 所示。

图 11.43　创建数据集 DataSet2

单击【确定】按钮，完成数据集 DataSet2 的创建，该数据集将返回指定 ID 的新闻信息。

(3) 在表格 table2 的第 3 个单元格中，插入一个 6 行 1 列的表格 talbe4，其属性设置如图 11.44 所示。

图 11.44　表格属性设置

(4) 分别设置表格第 2 行和第 5 行的单元格高度为 1，并设置其背景颜色值为“#000066”，这两行将显示为一条细的分隔线。同时，设置表格第 1 行和第 3 行的单元格水平对齐方式为居中对齐。

(5) 在【绑定】面板中选择并展开数据集 DataSet2，选择并拖动字段 ArtTitle 至表格 table4 的第 1 个单元格中，并在【属性】面板中设置其大小为 16px，字体颜色值为“#000066”，加粗显示。然后，拖动字段 AddTime 和 CopyFrom 至表格 table4 的第 3 行的单元格中，拖动字段 ArtContent 至表格 table4 的第 4 行的单元格中。最后，在表格第 6 行的单元格中输入文本“上一条：”和文本“下一条：”，如图 11.45 所示。

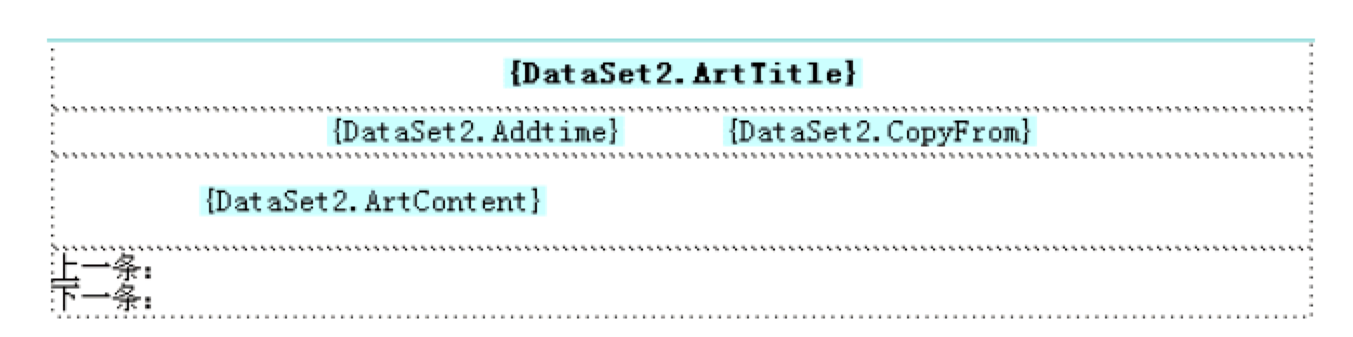


图 11.45　表格设计

(6) 在添加新闻信息时，可通过按 Enter 键换行或空格等来控制新闻的显示样式。但在显示为 HTML 页面时，这些特殊符号常常会被忽略，导致新闻显示的样式与设计时的样式完全不同。为此，需要自定义一个函数来格式化所要输出的新闻内容，将其中的某些特殊符号转换为相应的 HTML 标记。自定义的函数名为 ConvertHTML，其代码如下：

【示例代码】

```
Function ConvertHTML(ByVal Str1)
  Str1 = Replace(Str1, ">", ">")
  Str1 = Replace(Str1, "<", "<")
  Str1 = Replace(Str1, Chr(32), " ")
  Str1 = Replace(Str1, Chr(9), " ")
  Str1 = Replace(Str1, Chr(34), "'")
  Str1 = Replace(Str1, Chr(39), "'")
  Str1 = Replace(Str1, Chr(13), "")
  Str1 = Replace(Str1, Chr(10) & Chr(10),
"</P><P>        ")
  Str1 = Replace(Str1, Chr(10), "<BR>    ")
  ConvertHTML = Str1
End Function
```

(7) 在【服务器行为】面板中双击动态文本 DataSet2.ArtContent，在弹出的【动态文本】对话框中，将其代码修改为“<%# ConvertHTML (DataSet2.FieldValue("ArtContent",

Container)) %>”，如图 11.46 所示。

图 11.46　【动态文本】对话框

指定新闻信息的绑定已经完成，下面来看【上一条】动态链接与【下一条】动态链接的数据绑定。

(8) 在【服务器行为】面板中单击按钮，在弹出的菜单中选择【数据集】命令。单击【高级】按钮，切换至【数据集】对话框的高级模式。在【数据集】对话框的高级模式中，设置数据集的名称为 DataSet3，连接选择 Cnn。

(9) 单击【添加参数】按钮，在弹出的【添加参数】对话框中设置参数名称为“@Id”，并设置其值为 URL 参数 Id。重复以上操作，再次添加一个参数“@id1”，其值同样为 URL 参数 Id。

然后，在 SQL 文本框中输入以下 SQL 语句：

```
SELECT top 1 * FROM ArtInfo WHERE classid=(select classid from artinfo where
id=?) and addtime>(select addtime from artinfo where id=?) ORDER BY addtime
desc
```

如图 11.47 所示。

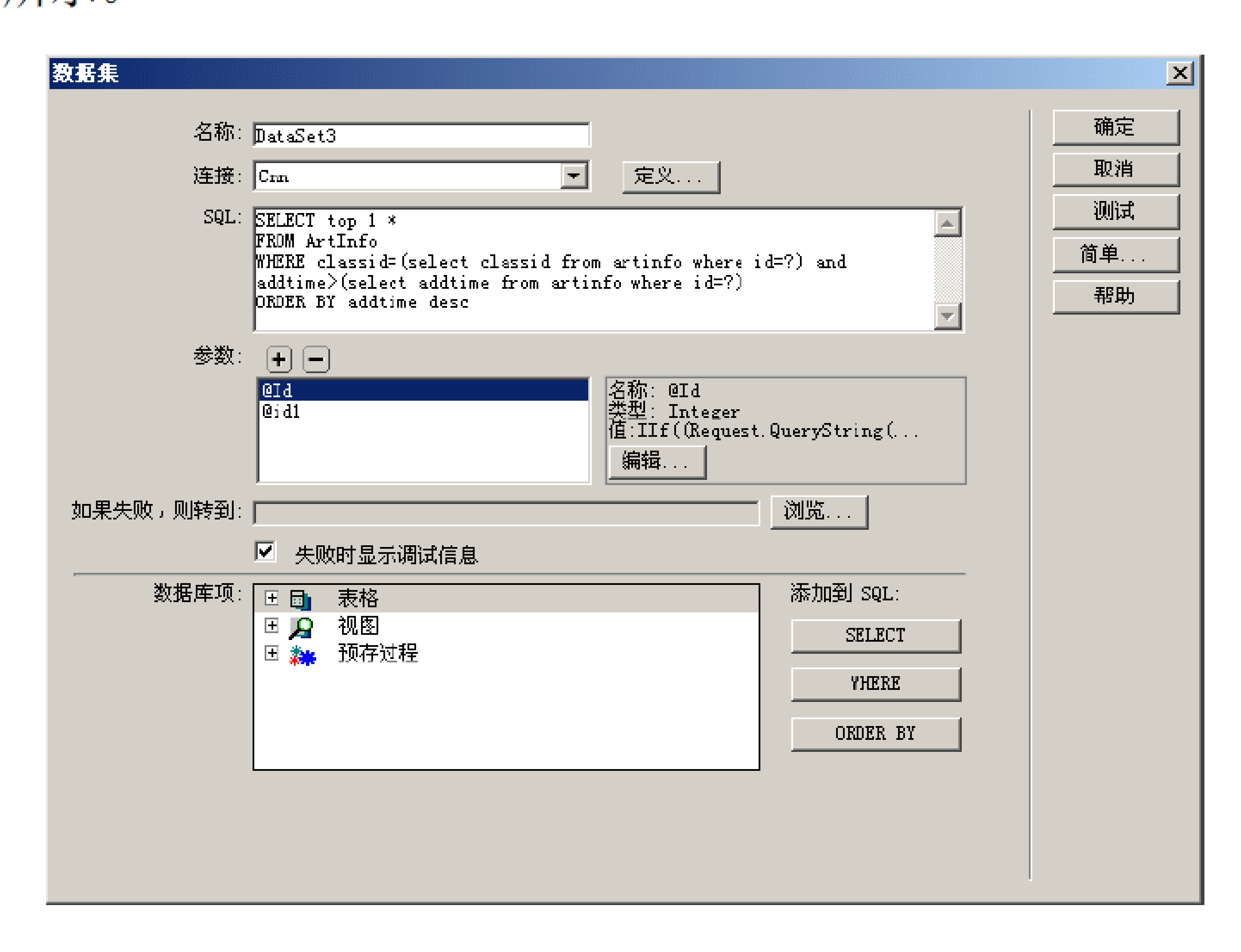


图 11.47　创建数据集 DataSet3

单击【确定】按钮，完成数据集 DataSet3 的创建，该数据集将返回上一条新闻的相关信息。

(10) 在【服务器行为】面板中，复制数据集 DataSet3 并粘贴生成新的数据集 DataSet4。双击数据集 DataSet4，将其 SQL 语句修改为：

```
SELECT top 1 * FROM ArtInfo WHERE classid=(select classid from artinfo where
id=?) and addtime<(select addtime from artinfo where id=?) ORDER BY addtime
desc
```

该数据集将返回下一条新闻的相关信息。

(11) 在【绑定】面板中选择并展开数据集 DataSet3，拖动字段 ArtTitle 至文本"上一条："之后，并在【属性】面板中设置其链接为"Art_Show.aspx?id=<%# DataSet3.FieldValue("id", Container) %>"。在该动态文本之后，输入文本"没有了"，并设置其字体颜色为红色。选择该文本，在【服务器行为】面板中，单击+按钮，在弹出菜单中选择【显示区域】|【数据集为空时显示】命令，并在弹出的对话框中选择数据集 DataSet3，如图 11.48 所示。

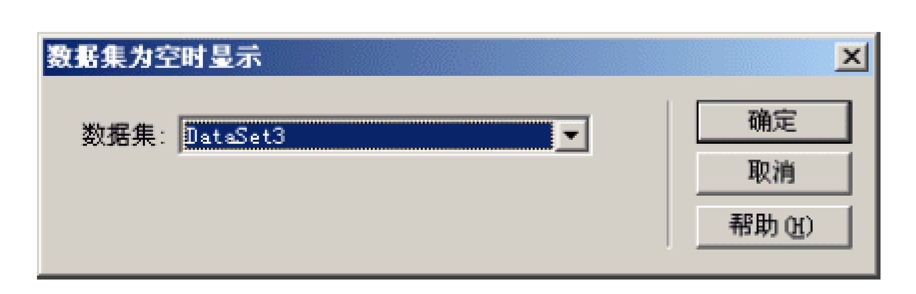


图 11.48　【数据集为空时显示】对话框

(12) 重复以上操作，选择并展开数据集 DataSet4，拖动字段 ArtTitle 至文本"下一条："之后，并在【属性】面板中设置其链接为"Art_Show.aspx?id=<%# DataSet4.FieldValue("id", Container) %>"。在该动态文本之后输入文本"没有了"，并选择该文本，按步骤(11)打开【数据集为空时显示】对话框，并在弹出的对话框中选择数据集 DataSet4。

(13) 最后，还需修改页面的 Page_Load 事件，以便更新指定新闻的点击数，其代码如下：

【示例代码】

```
Sub Page_Load(Sender As Object,E As EventArgs)
  Dim Cnn As OleDbConnection
  Dim Cmd As OleDbCommand
  Dim strConn As String
  Dim Sql As String
  If Not IsPostBack Then
    LoadClass()  '加载新闻分类树形菜单
    '获取数据库连接字符串
    strConn
=System.Configuration.ConfigurationSettings.AppSettings("MM_CONNECTION_S
TRING_Cnn")
    Cnn = New OleDbConnection(strConn)
    Cnn.Open()  '打开数据库连接
    '将指定新闻的 ClickNum 加 1
    Sql = "update ArtInfo Set clicknum=clicknum+1 where id=" & trim(request.
QueryString("id"))
    Cmd = New OleDbCommand(Sql, Cnn)
    Cmd.ExecuteNonquery
```

```
    Cnn.Close()  '关闭数据库连接
  End If
End Sub
```

至此，新闻阅读页面的功能全部设计完成，页面预览效果如图 11.49 所示。

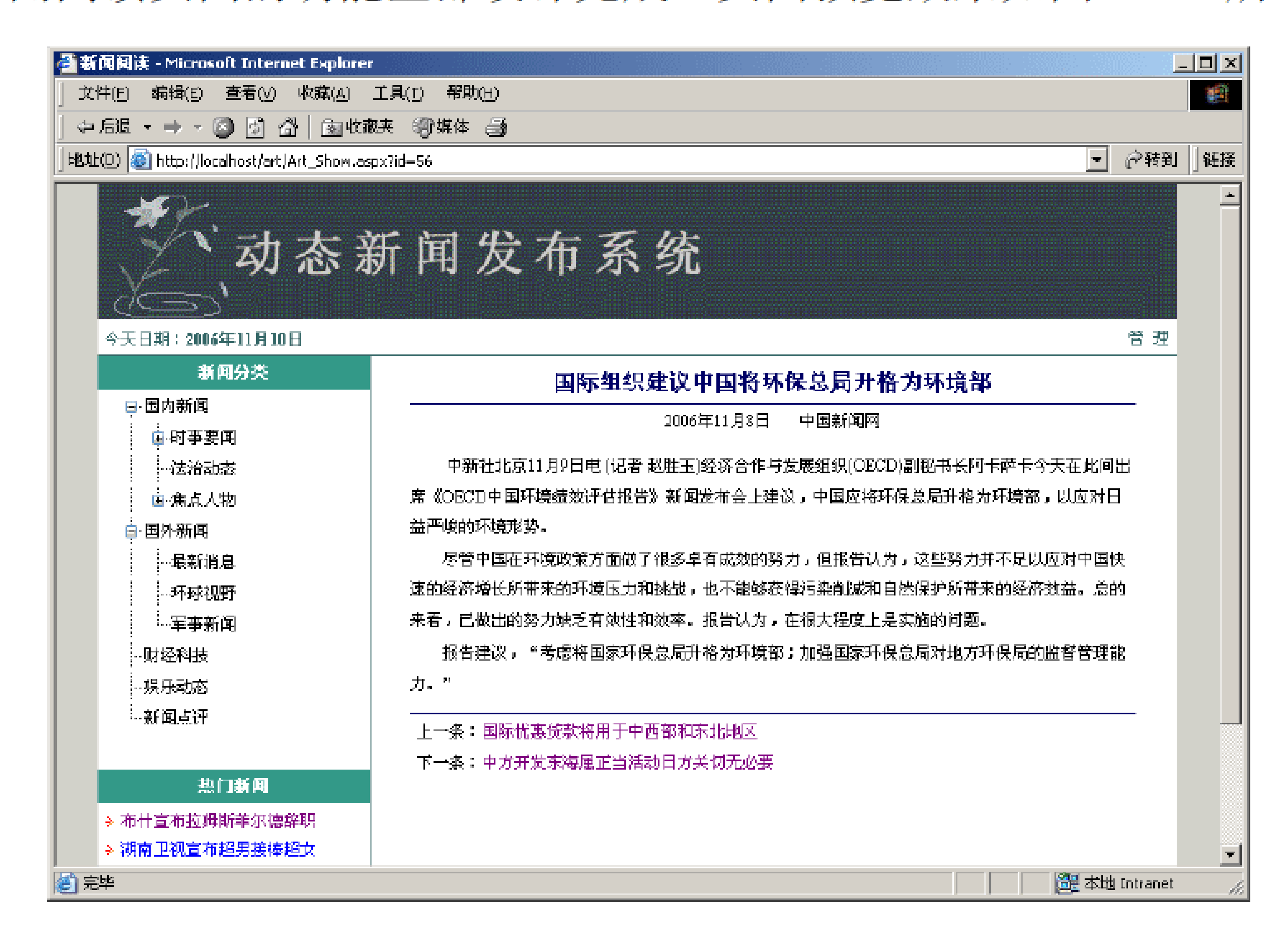


图 11.49　页面预览效果

11.5　新 闻 管 理

新闻管理是新闻发布系统后台模块的一个功能，仅管理员可以执行其操作。正常情况下，管理员登录后方可执行后台操作。这里为了用更多篇幅介绍这一功能，就跳过了管理员的登录页面的介绍。当用户单击系统主页面中的【管理】链接时，将直接进入新闻管理的后台操作页面。

11.5.1　新闻管理

(1) 新建一个 ASP.NET VB 类型的页面，将其命名为 xwgl.aspx。选择【修改】|【模板】|【套用模板到页】命令，在弹出的【选择模板】对话框中选择【后台模板】，将模板应用到本页面。

首先，创建一个数据集，用于绑定动态新闻信息。

(2) 在【服务器行为】面板中单击 + 按钮，在弹出的菜单中选择【数据集】。单击【高级】按钮，切换至【数据集】对话框的高级模式下。

(3) 设置数据集的名称为 DataSet1，选择连接 Cnn，并在 SQL 文本框中输入以下语句：

```
SELECT artinfo.id,arttitle,classname,author,copyfrom,addtime,clicknum,sh
FROM artinfo,classinfo WHERE artinfo.classid=classinfo.classid ORDER BY
addtime desc
```

如图 11.50 所示。

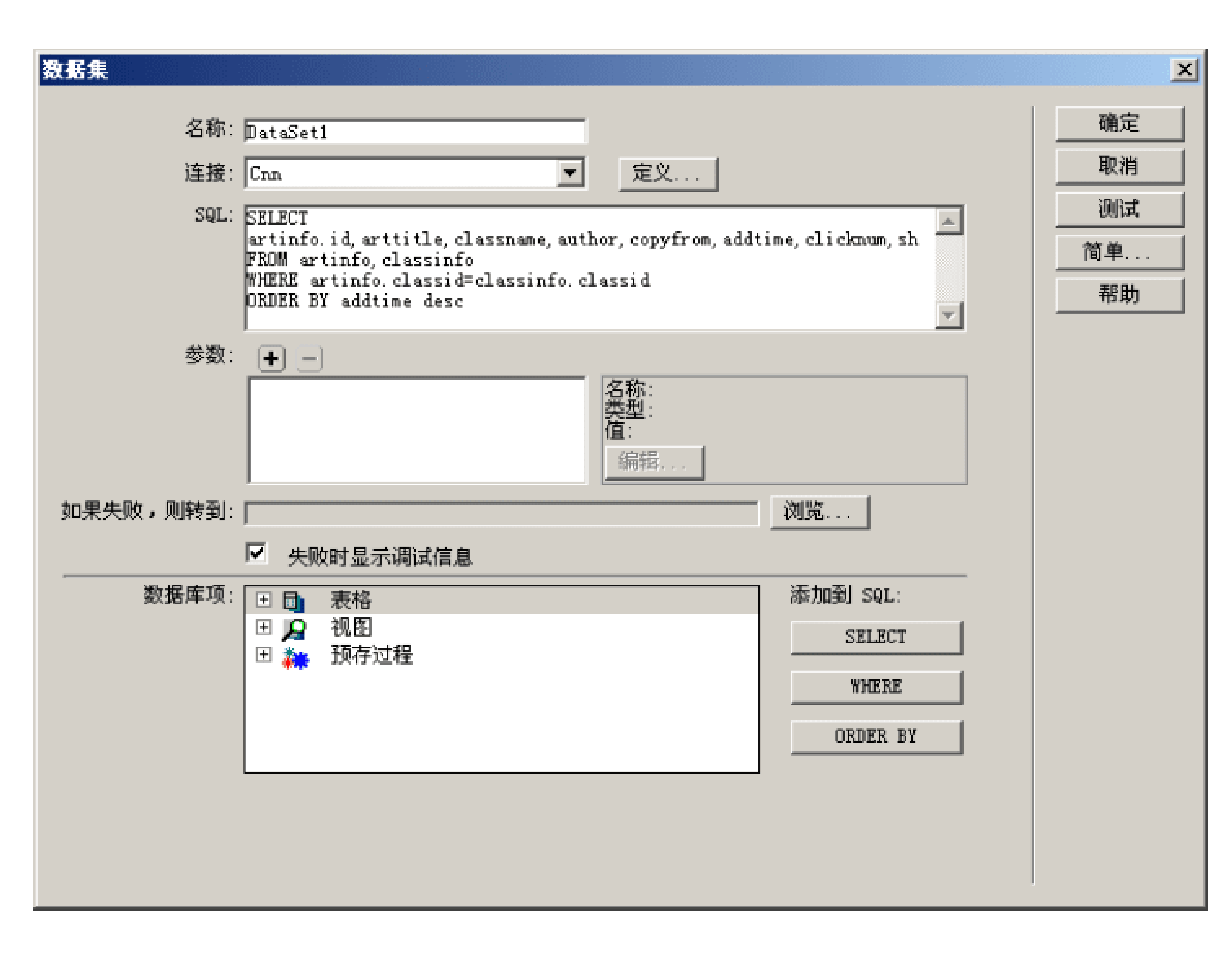


图 11.50　创建数据集 DataSet1

单击【确定】按钮，完成数据集 DataSet1 的创建，该数据集将返回所有的新闻信息，包括已经审核的和未审核的(在前台模块的页面中，均只显示已经审核的新闻信息)。

(4) 将光标置于表格 table2 的第 3 个单元格中，在【服务器行为】面板中单击+按钮，从弹出的菜单中选择【数据网格】命令。在弹出的【数据网格】对话框中，设置 ID 为 DataGrid1，数据集选择 DataSet1，并设置为一次显示 10 条记录，在【导航】下拉列表框中选择【编号链接到每一页】，如图 11.51 所示。

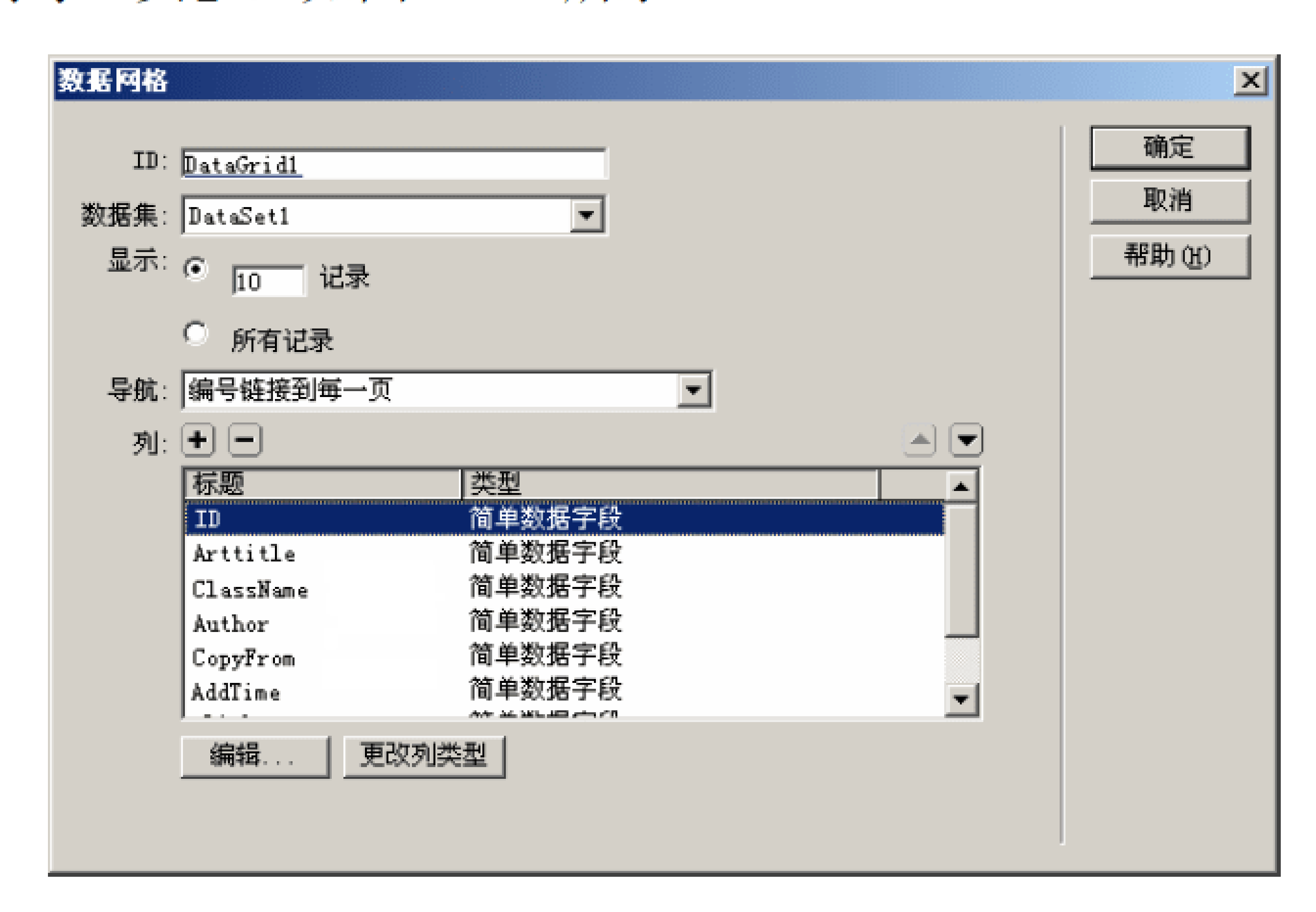


图 11.51　创建数据网格 DataGrid1

(5) 通过单击【删除列】按钮−，在【列】列表框中删除不需要在数据网格中显示的列 ID 和 Author。然后，分别设置列 ArtTitle、ClassName、CopyFrom、AddTime 和 ClickNum 的显示标题为“新闻标题”、“所属分类”、“新闻来源”、“发布时间”和“点击数”。

(6) 选择列 sh，单击【更改列类型】按钮，从弹出的菜单中选择【自由格式】命令。在弹出的【自由格式列】对话框中，设置其标题为“是否审核”。单击【添加数据字段】按钮，在弹出的【添加数据字段】对话框中选择字段 SH，如图 11.52 所示。

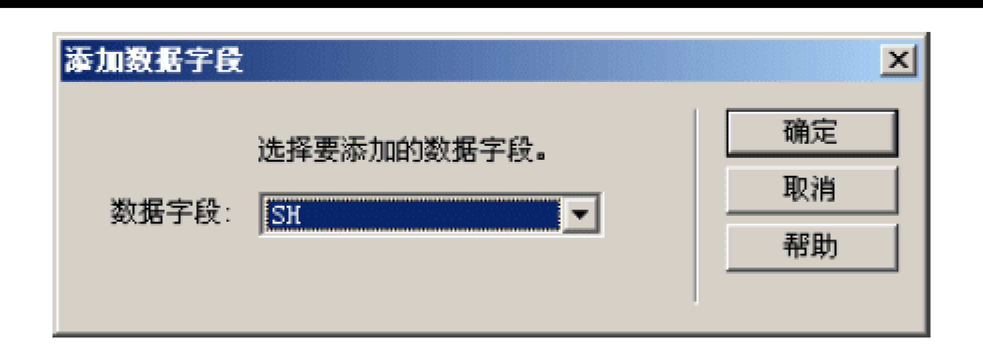


图 11.52　【添加数据字段】对话框

(7) 单击【确定】按钮，返回【自由格式列】对话框，并在【内容】文本框中将所添加的数据字段修改为“<%# IIf((DataSet1.FieldValue("sh", Container)=1),"已审","未审") %>”，如图 11.53 所示。

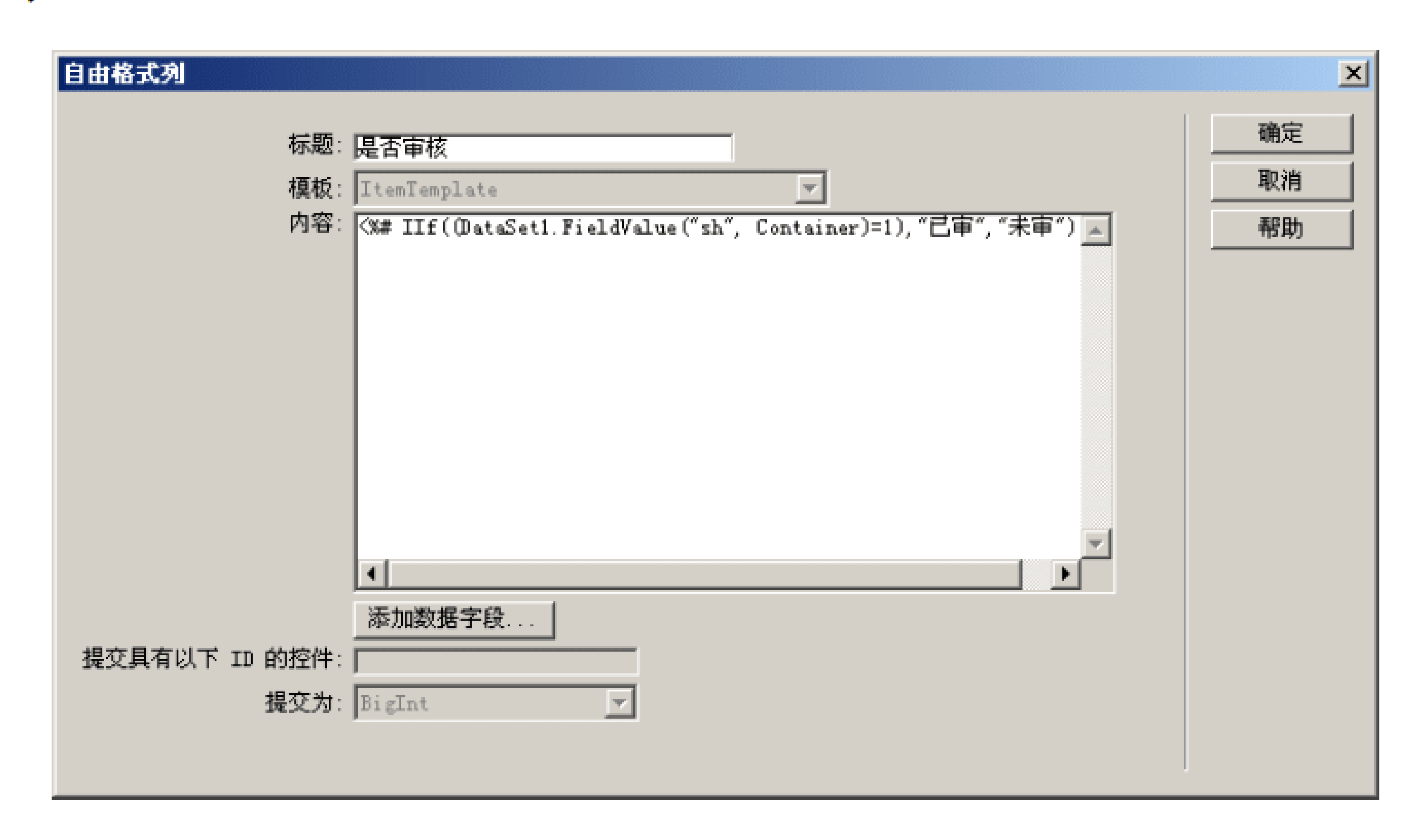


图 11.53　【自由格式列】对话框

单击【确定】按钮，完成自由格式列的设置。

(8) 单击【添加列】按钮+，从弹出的菜单中选择【超级链接】命令。在弹出的【超级链接列】对话框中设置【标题】为“编辑”，【超级链接文本】为静态文本“编辑”，【链接页】为数据字段 ID，并设置【格式字符串】为“Art_Edit.aspx?id={0}”，如图 11.54 所示。

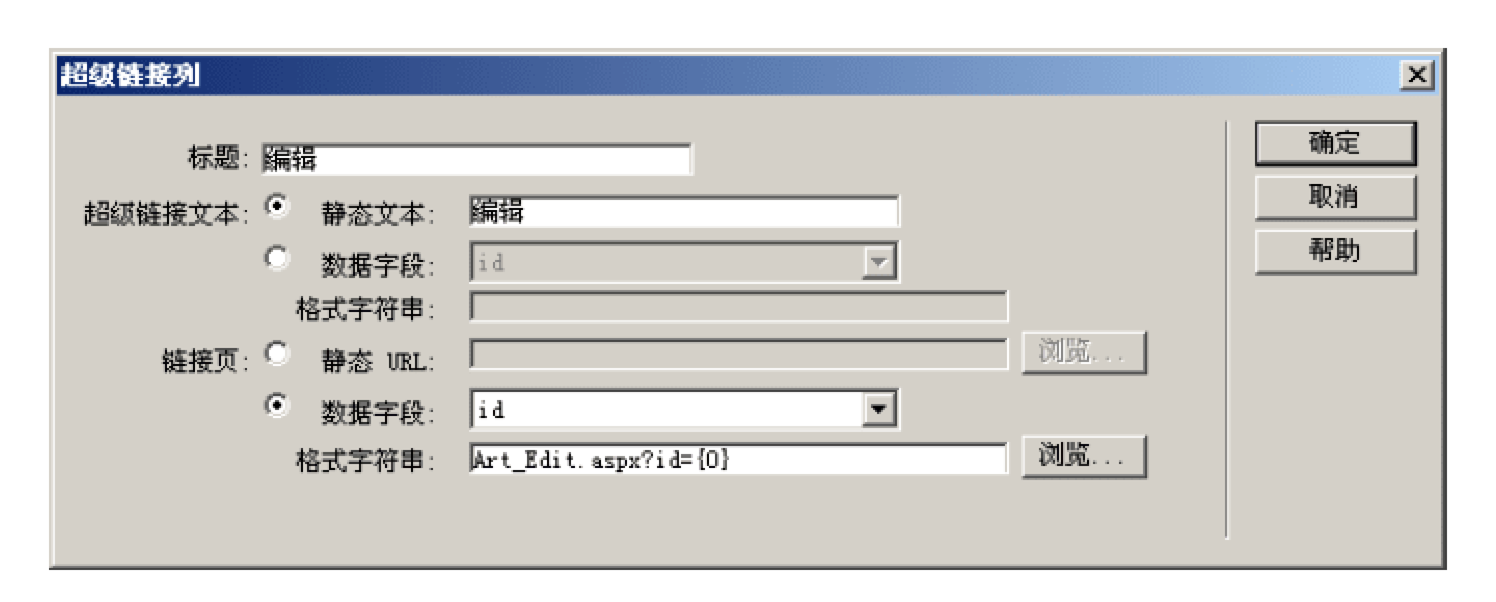


图 11.54　【超级链接列】对话框

单击【确定】按钮，完成超级链接列的添加。

(9) 单击【添加列】按钮+，从弹出的菜单中选择【删除】命令。在弹出的【删除按钮列】对话框中，设置【标题】为“删除”，【按钮类型】为【链接按钮】，【删除自】为 ArtInfo，主键为 ID，如图 11.55 所示。

图 11.55　【删除按钮列】对话框

(10) 单击【确定】按钮，完成删除按钮列的添加，也完成了对列的设置。此时，【数据网格】对话框如图 11.56 所示。

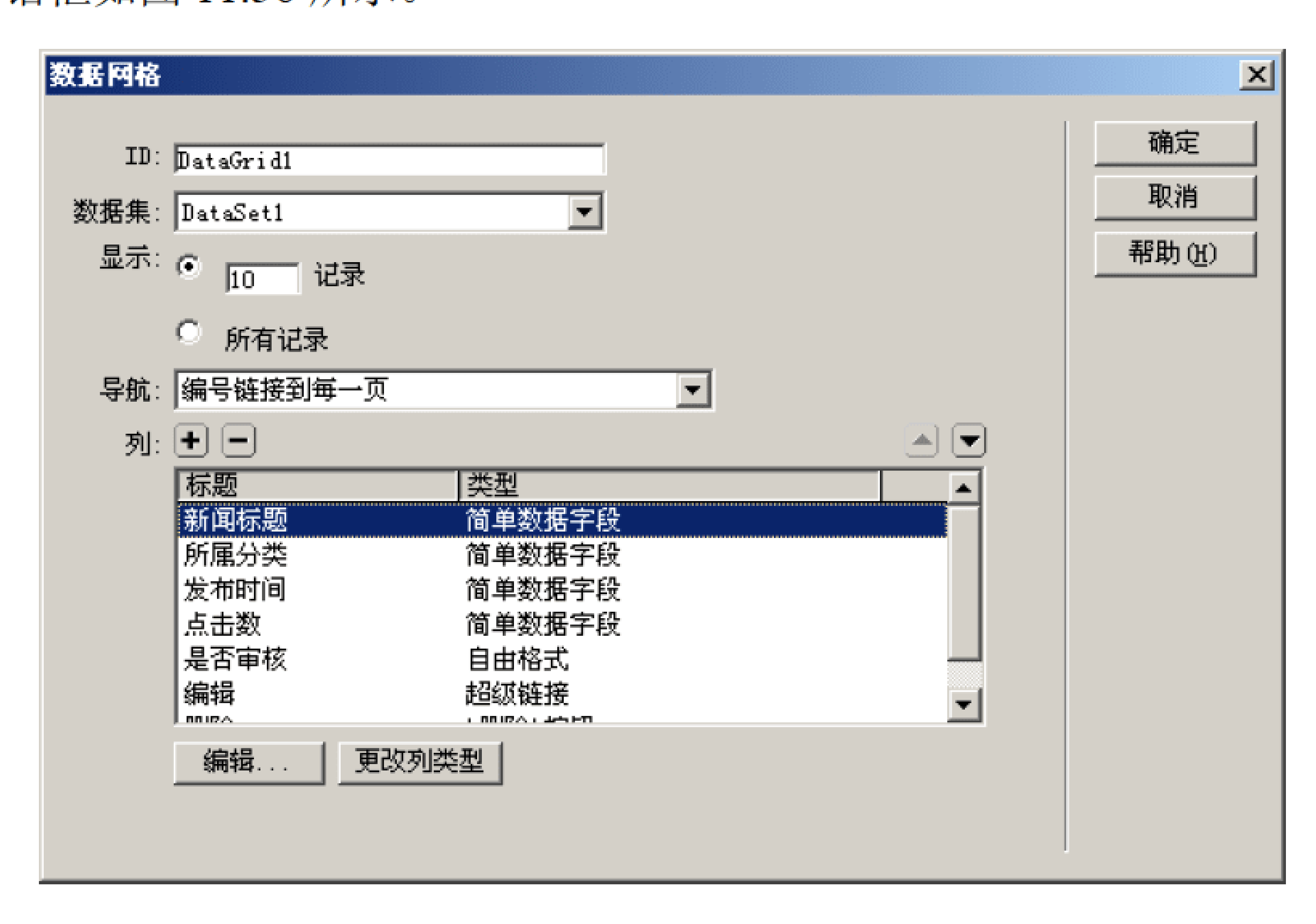


图 11.56　数据网格 DataGrid1

单击【确定】按钮，完成数据网格 DataGrid1 的创建。

这里，没有采用传统的编辑方式来对指定的新闻信息进行修改。因为，对于新闻内容来说，不可能显示在数据网格中。如果采用传统的直接编辑方式，就无法对新闻内容进行修改。因此，这里将【编辑】按钮转变为一个超级链接，将其跳转至新闻编辑页面 Art_Edit.aspx 中进行操作，同时将当前的新闻 ID 作为页面参数进行传递。

此外，对于【是否审核】列来说，其在数据库中存储的数据为 1(表示已审核)和 0(表示未审核)。如果直接将这些数据显示在数据网格中，用户将无法理解。因此，这里采用【自由格式列】对话框自定义该列的数据显示。在自定义的内容中，调用了 IIF 函数来对字段 sh 的值进行判断，如果其值为 1，则返回“已审”；否则，返回“未审”。

至此，新闻管理的功能页面设计完成，页面预览效果如图 11.57 所示。

单击【删除】按钮，可直接删除指定的新闻信息；单击【编辑】按钮，则将跳转至新闻编辑页面 Art_Edit.aspx 中进行编辑操作。

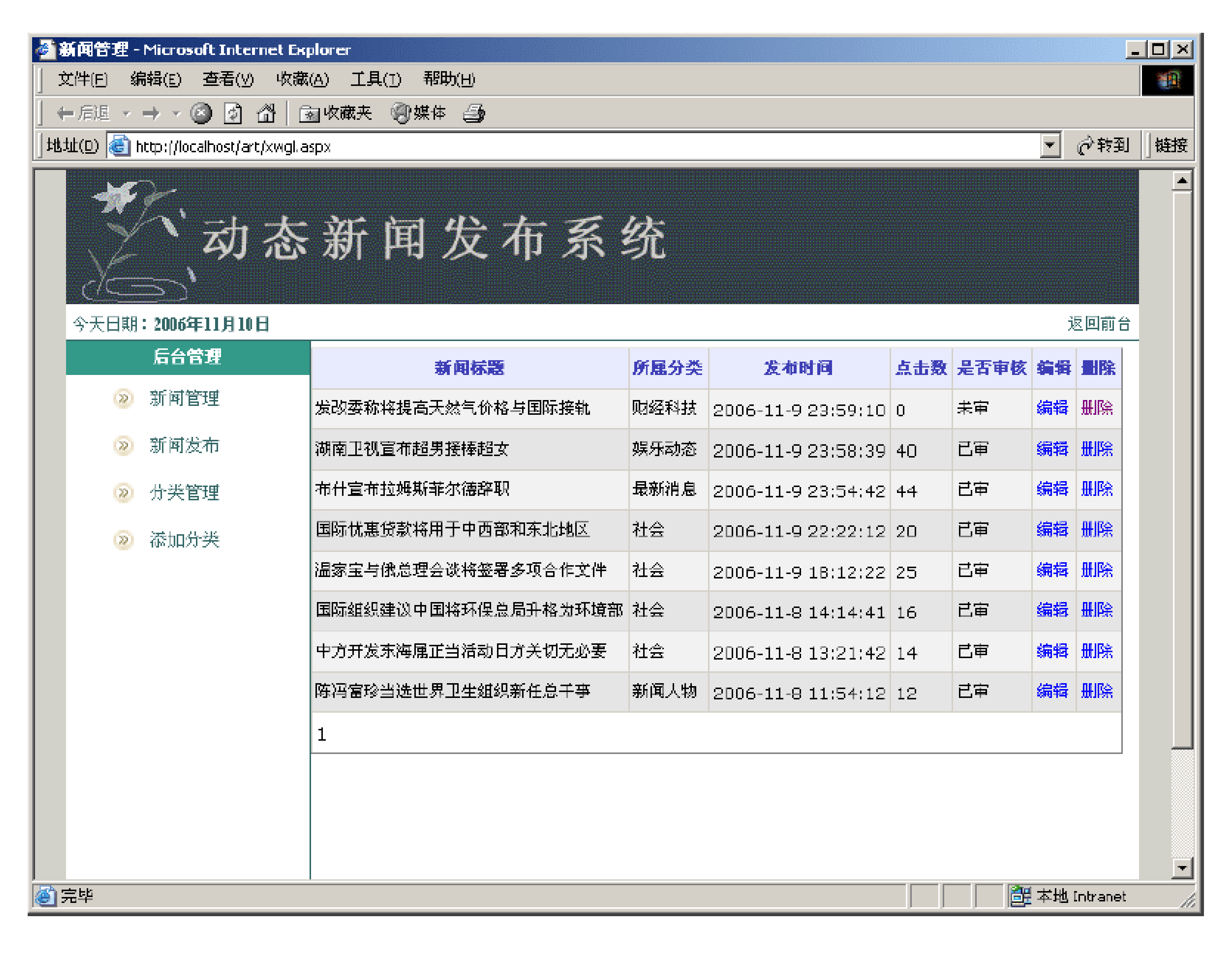


图 11.57　页面预览效果

11.5.2　发布新闻

发布新闻页面的功能主要是添加一条新的新闻信息。

(1)　新建一个 ASP.NET VB 类型的页面，将其命名为 Art_Add.aspx。选择【修改】|【模板】|【套用模板到页】命令，在弹出的【选择模板】对话框中选择【后台模板】，将模板应用到本页面。

(2)　在表格 table2 的第 3 个单元格中，插入一个 5 行 6 列的表格 table3，其属性设置如图 11.58 所示。

图 11.58　表格属性设置

(3)　在表格 table3 的各个单元格中输入相应的文本，并添加用于信息输入的各个 Web 服务器控件，如图 11.59 所示。

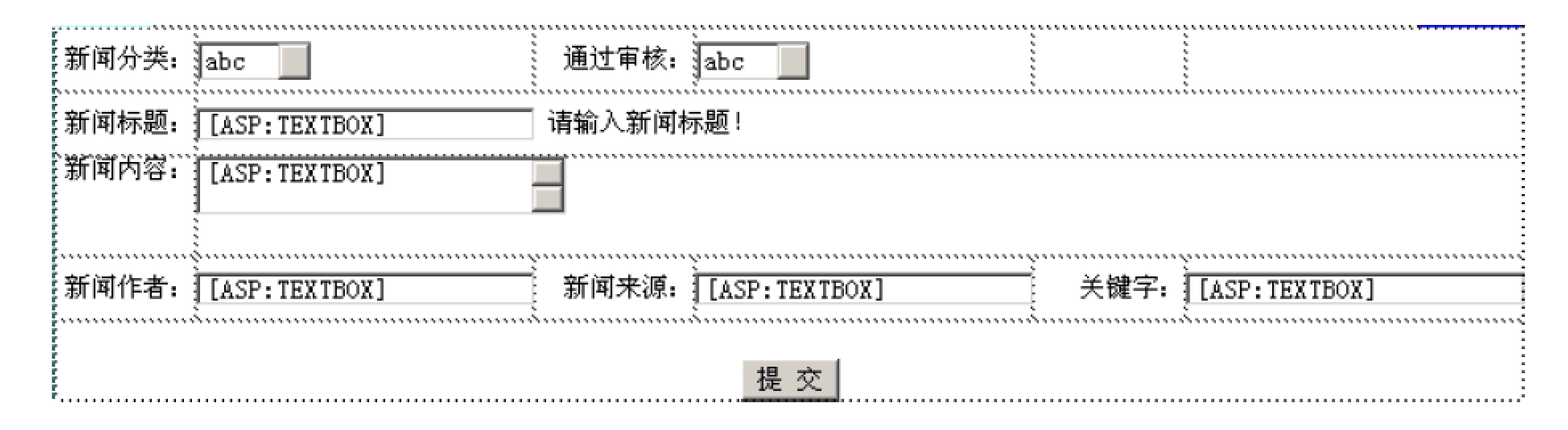


图 11.59　表格设计

其中，通过审核对应的下拉列表框中，添加了两个数据项，其显示文本分别为“已审核”和“未审核”，返回值分别为 1 和 0，其定义代码如图 11.60 所示。

```
<td width="21%" align="left">
  <asp:DropDownList id="sh" runat="server">
    <asp:ListItem Value="1">已审核</asp:ListItem>
    <asp:ListItem Value="0">未审核</asp:ListItem>
  </asp:DropDownList>
</td>
```

图 11.60　【通过审核】下拉框控件

在新闻标题所对应的文本框之后，添加了一个 RequiredFieldValidator 服务器控件，用于对新闻标题的输入进行验证，其属性设置如图 11.61 所示。

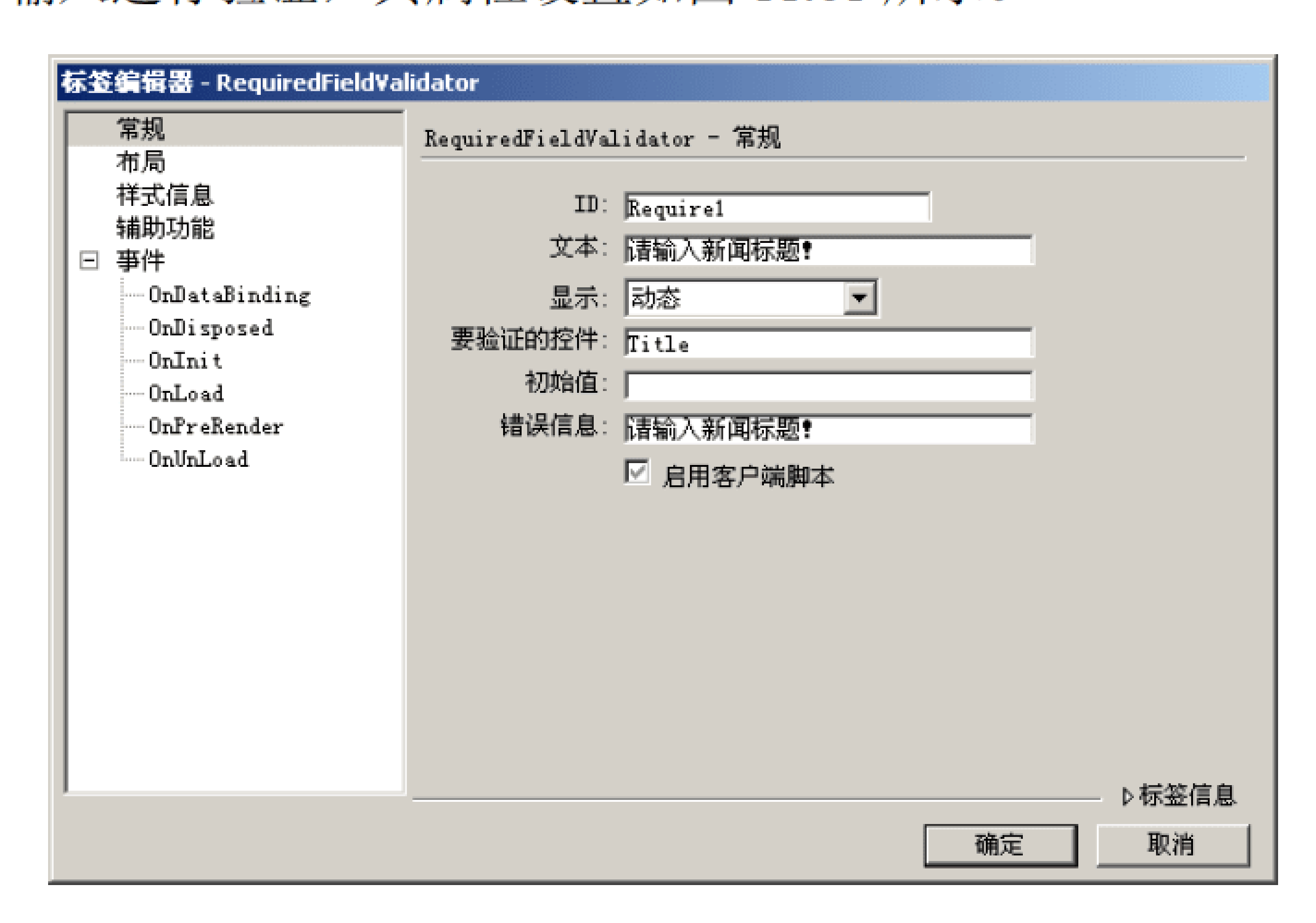


图 11.61　Require1 验证控件

对于【新闻分类】下拉列表框控件的数据绑定，是在代码中通过自定义过程 LoadClass1()来实现的。该过程代码在新闻搜索中已经进行了描述，这里不再赘述。

(4)　打开【服务器行为】面板，单击按钮，在弹出的菜单中选择【插入记录】命令。此时，将弹出【插入记录】对话框，如图 11.62 所示。

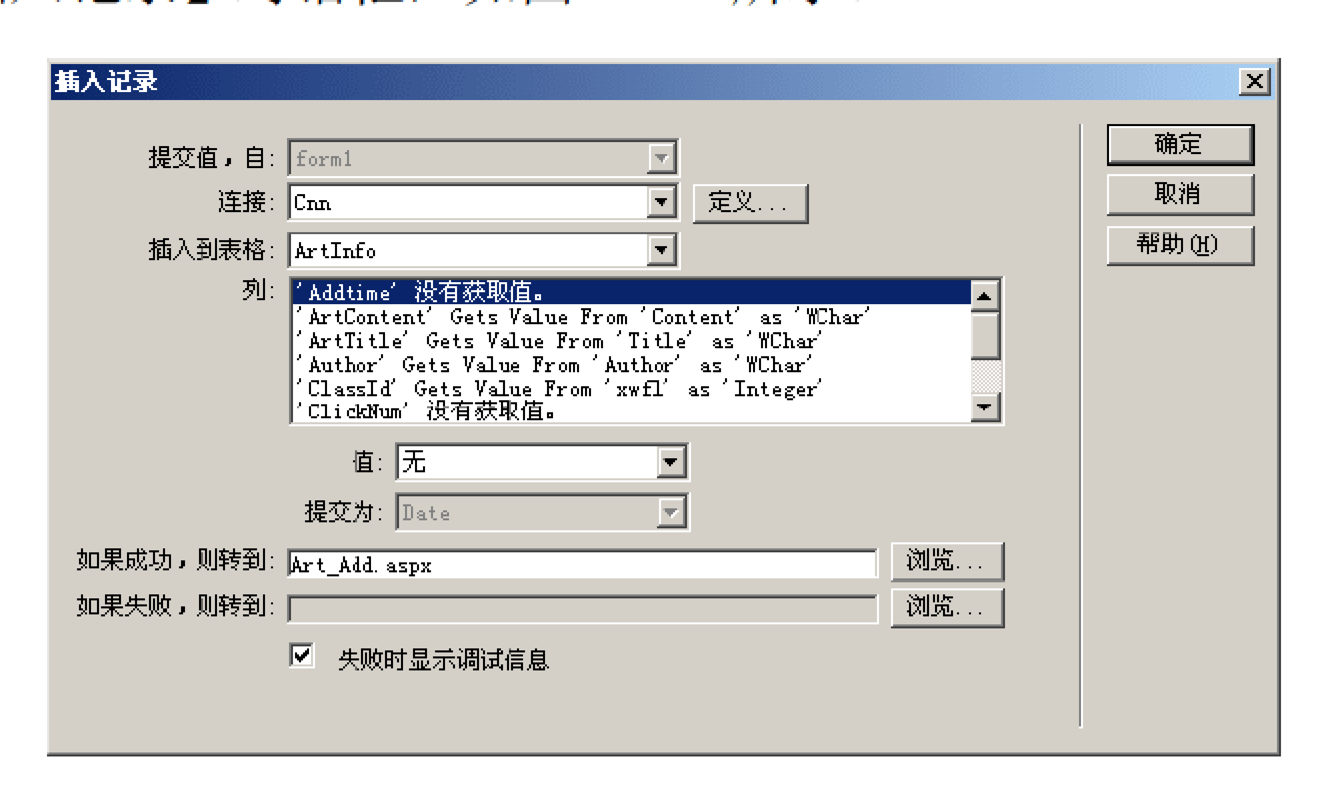


图 11.62　【插入记录】对话框

(5)　数据库连接选择 Cnn，选择表格 ArtInfo。然后，分别设置字段 ArtContent、ArtTitle、Author、ClassID、CopyFrom、KeySt 和 SH 的获取值为表单元素 Content、Title、Author、

Xwfl、CopyFrom、KeyWord 和 SH 的值。最后，设置当数据提交成功后转向的页面为 Art_Add.aspx，即仍返回本页面。

(6) 单击【确定】按钮，完成插入记录的创建。

至此，发布新闻页面的功能设计完成，其页面预览效果如图 11.63 所示。

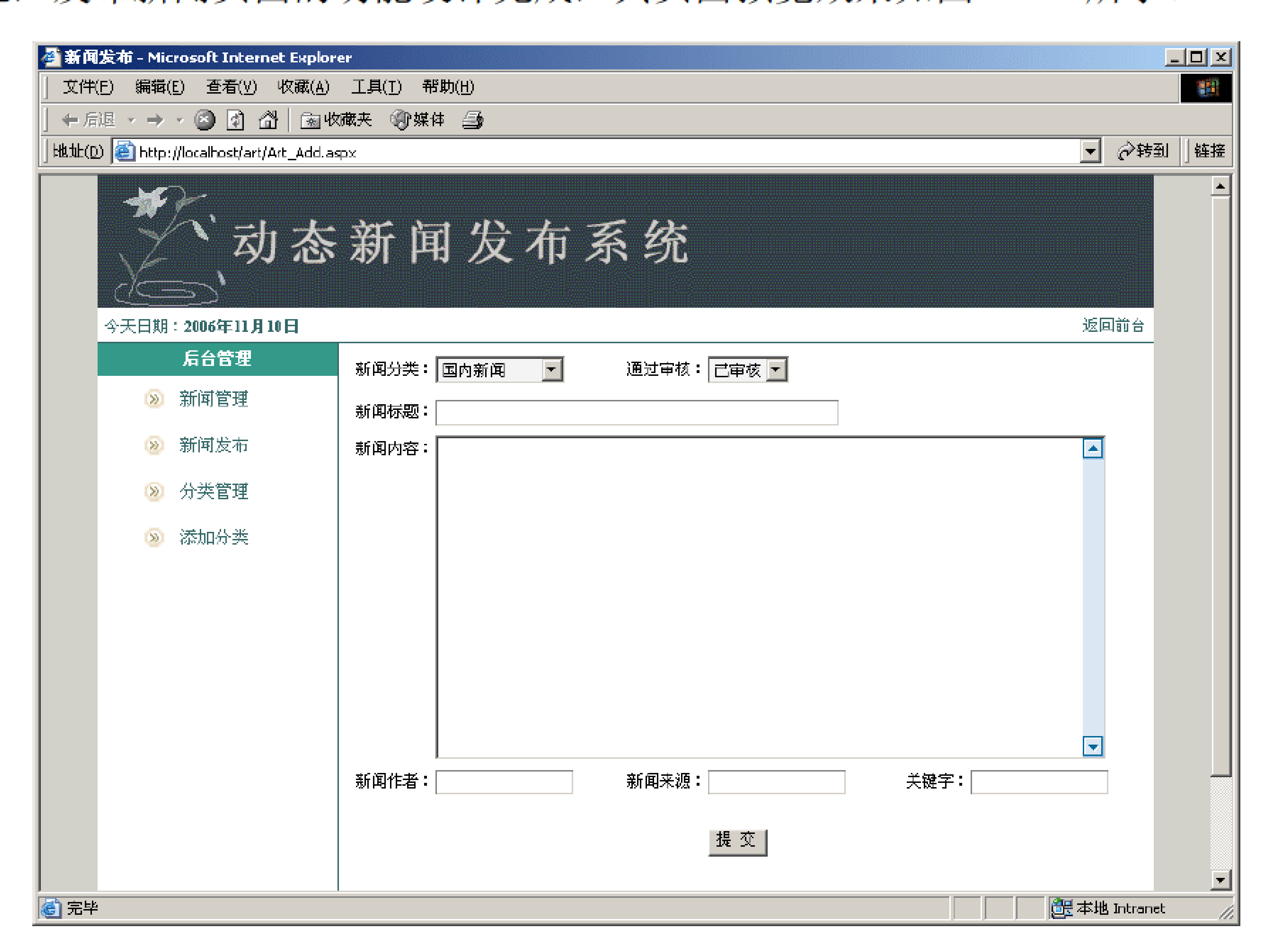


图 11.63　页面预览效果

11.6　分 类 管 理

11.6.1　分类管理

分类管理主要用于对新闻的分类信息进行管理，其操作包括对分类信息的编辑和删除。

(1) 新建一个 ASP.NET VB 类型的页面，将其命名为 Class_Admin.aspx。选择【修改】|【模板】|【套用模板到页】命令，在弹出的【选择模板】对话框中选择【后台模板】，将模板应用到本页面。

为了提供对分类信息的浏览，首先需要创建一个数据集。

(2) 打开【服务器行为】面板，单击按钮，从弹出的菜单中选择【数据集】命令。在弹出的【数据集】对话框中设置数据集名称为 DataSet1，连接选择 Cnn，表格选择 ClassInfo，排序条件设置为按字段 UpClass 升序排序，如图 11.64 所示。

单击【确定】按钮，完成数据集 DataSet1 的创建，该数据集将返回所有新闻分类信息。

(3) 将光标置于表格 table2 的第 3 个单元格中，在【服务器行为】面板中，单击按钮，从弹出的菜单中选择【数据网格】命令。在弹出的【数据网格】对话框中设置其 ID 为 DataGrid1，数据集为 DataSet1，设置一次显示 10 条记录，在【导航】下拉列表框中选择【编号链接到每一页】，如图 11.65 所示。

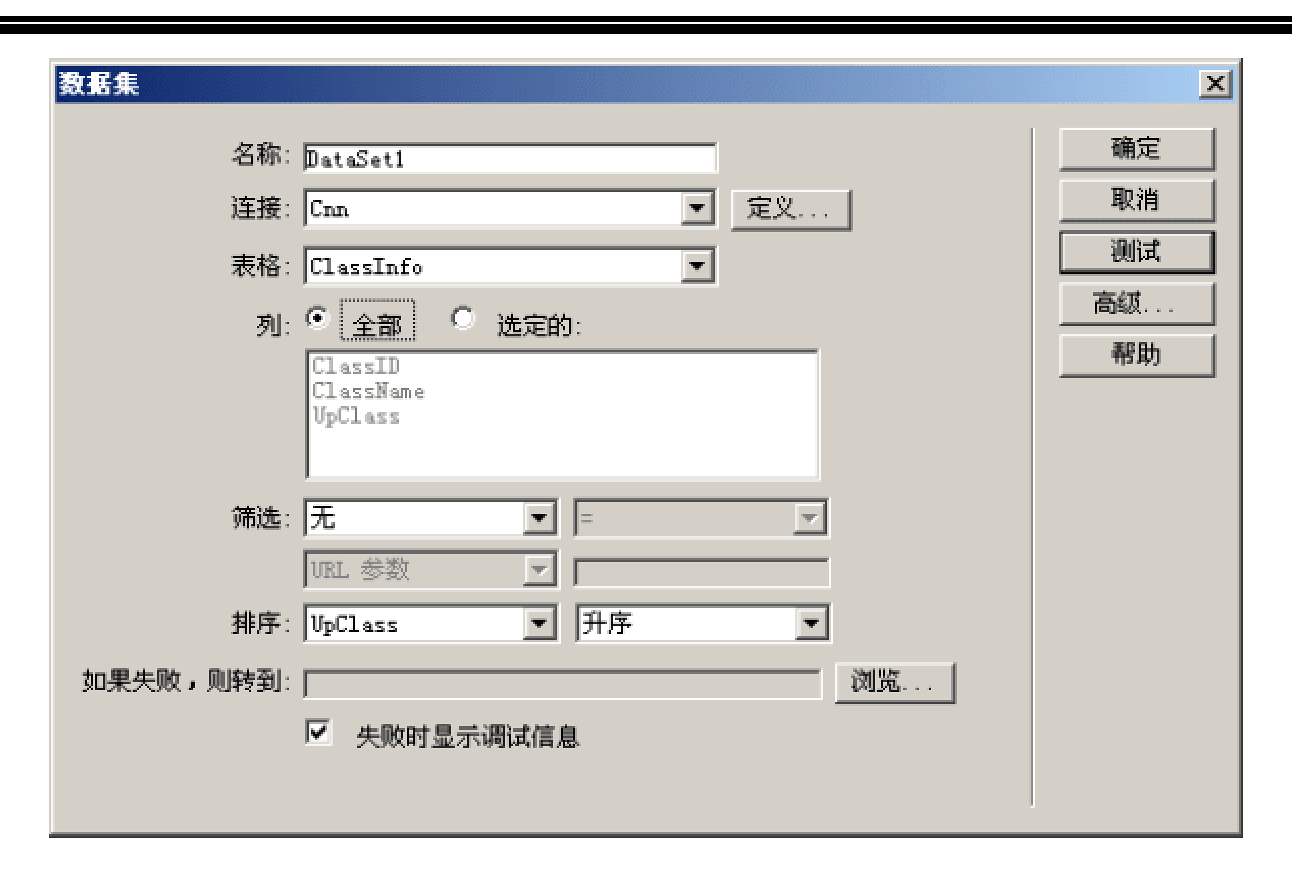


图 11.64　创建数据集 DataSet1

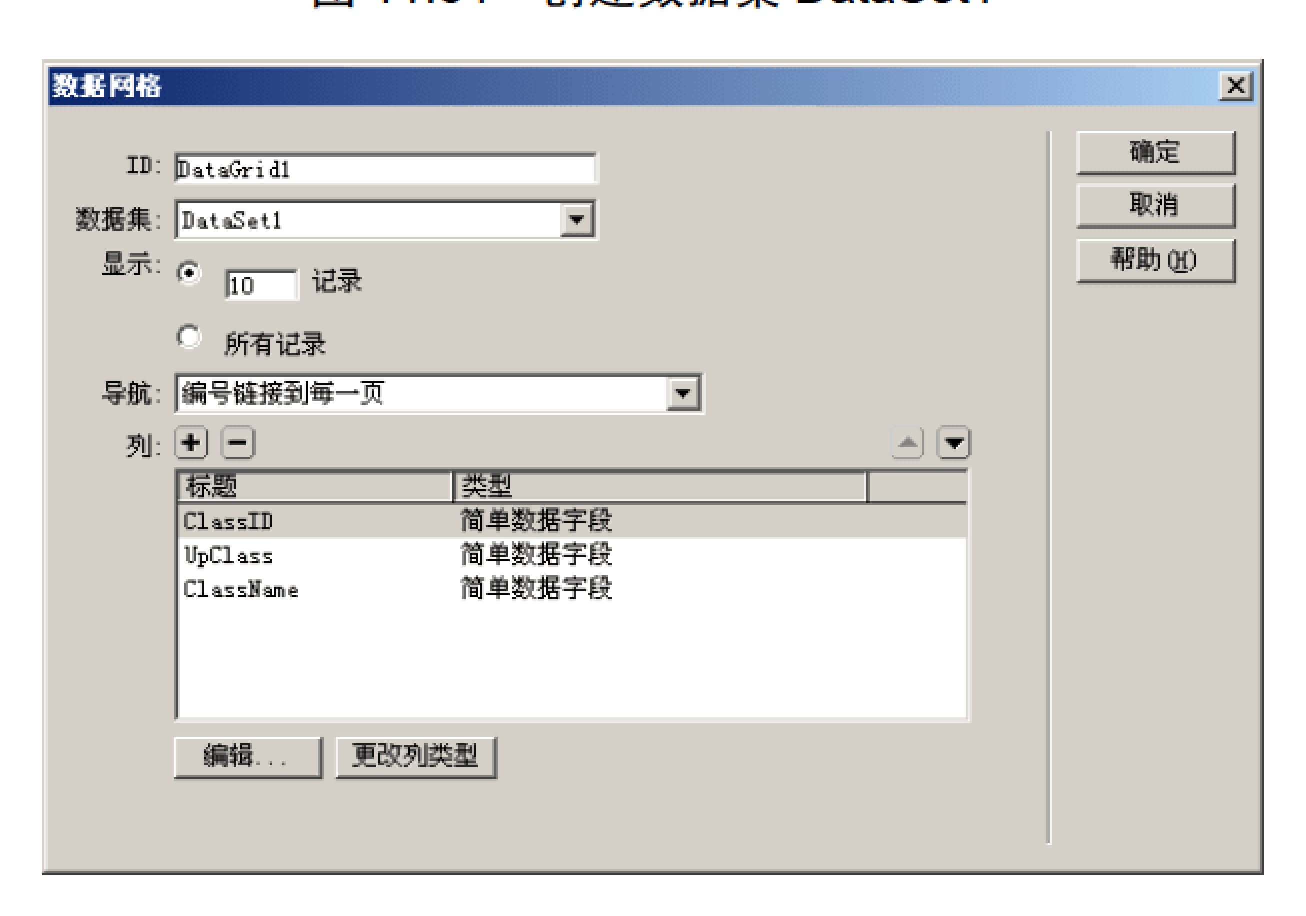


图 11.65　数据网格 DataGrid1

(4) 单击【添加列】按钮+，从弹出的菜单中选择【'编辑'、'更新'、'取消'按钮列】命令，并在弹出的对话框中设置【标题】为“编辑”，【按钮类型】为【链接按钮】，【更新表格】为 ClassInfo，【主键】为 ClassID，如图 11.66 所示。

图 11.66　【编辑、更新、取消按钮列】对话框

单击【确定】按钮，完成该按钮列的添加。

(5) 再次单击【添加列】按钮+，从弹出的菜单中选择【'删除'按钮】命令，并在弹出的对话框中设置【标题】为“删除”，【按钮类型】为【链接按钮】，【更新表格】为 ClassInfo，【主键】为 ClassID，如图 11.67 所示。

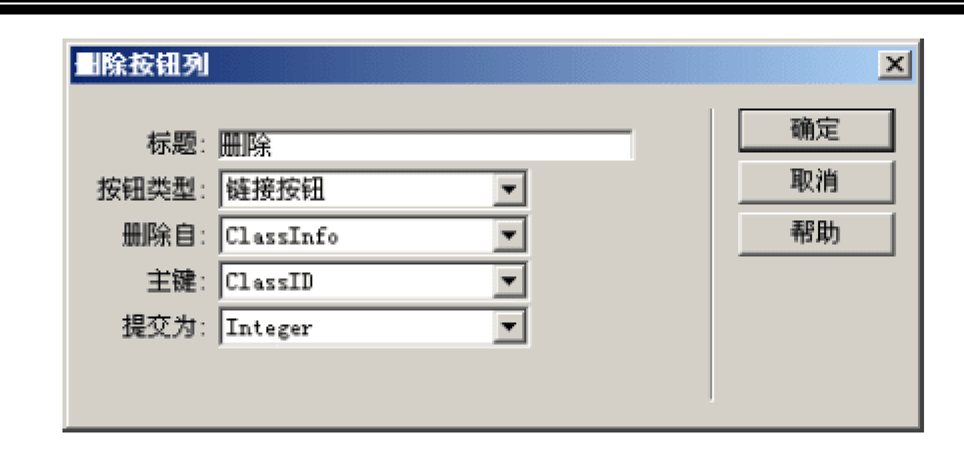


图 11.67　【删除按钮列】对话框

单击【确定】按钮，完成删除按钮列的添加。

(6) 依次选择列 ClassID、UpClass 和 ClassName，分别设置其显示标题为“分类编号”、“上级分类”和“分类名称”。其中，设置 ClassName 列时，需取消其只读属性，并设置提交值为 VarWChar，如图 11.68 所示。

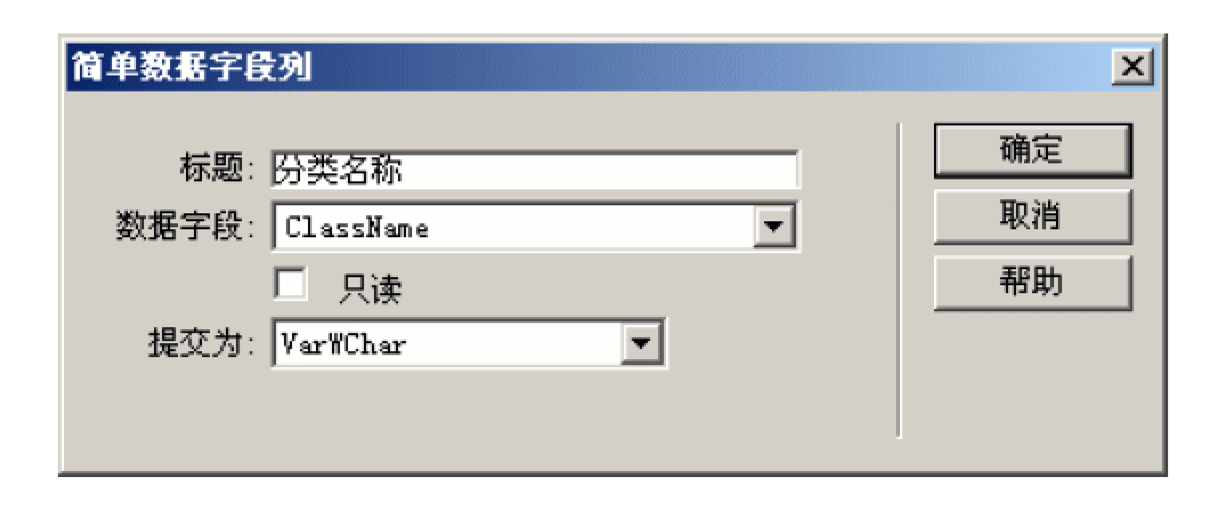


图 11.68　设置 ClassName 列

(7) 通过单击▲和▼按钮，调整字段列的顺序(从上到下)为“分类编号”、“分类名称”、“上级分类”、“编辑”和“删除”，如图 11.69 所示。

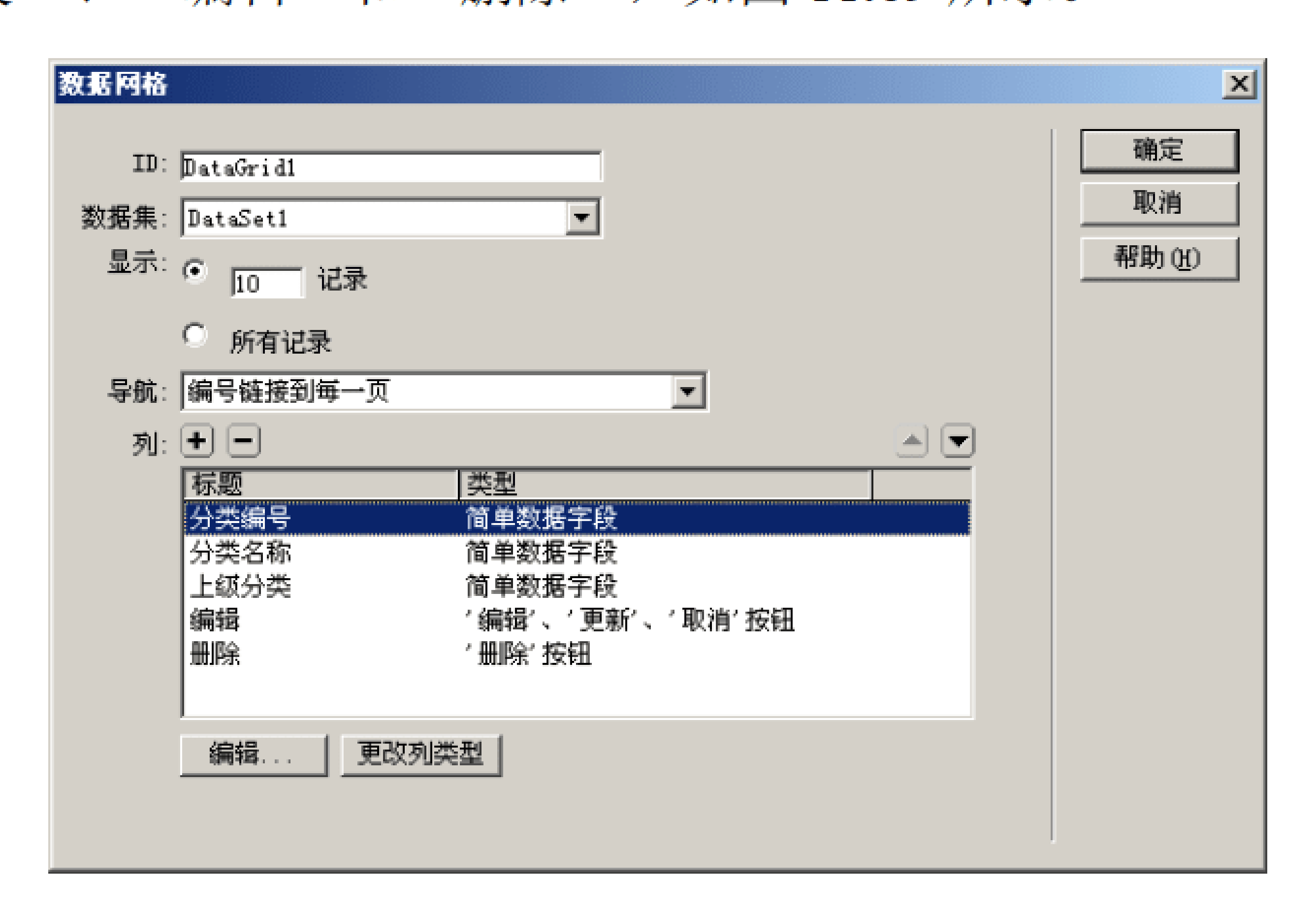


图 11.69　数据网格 DataGrid1

单击【确定】按钮，完成数据网格的创建。

由于在删除分类信息时，不能直接删除指定的分类，需先判断其是否存在与该分类相关的新闻信息以及该分类下是否存在子分类信息。因此，这里不能采用数据网格中默认的删除事件，而需自定义删除操作。

(8) 在【设计】视图中，选择数据网格 DataGrid1 并右击，从弹出的快捷菜单中选择【编辑标签】命令，并在弹出的【标签编辑器】对话框中选择【事件】|OnItemCommand，将其默认值 DataSet1.OnDataGridDelete 修改为 DataGrid1_Del，如图 11.70 所示。

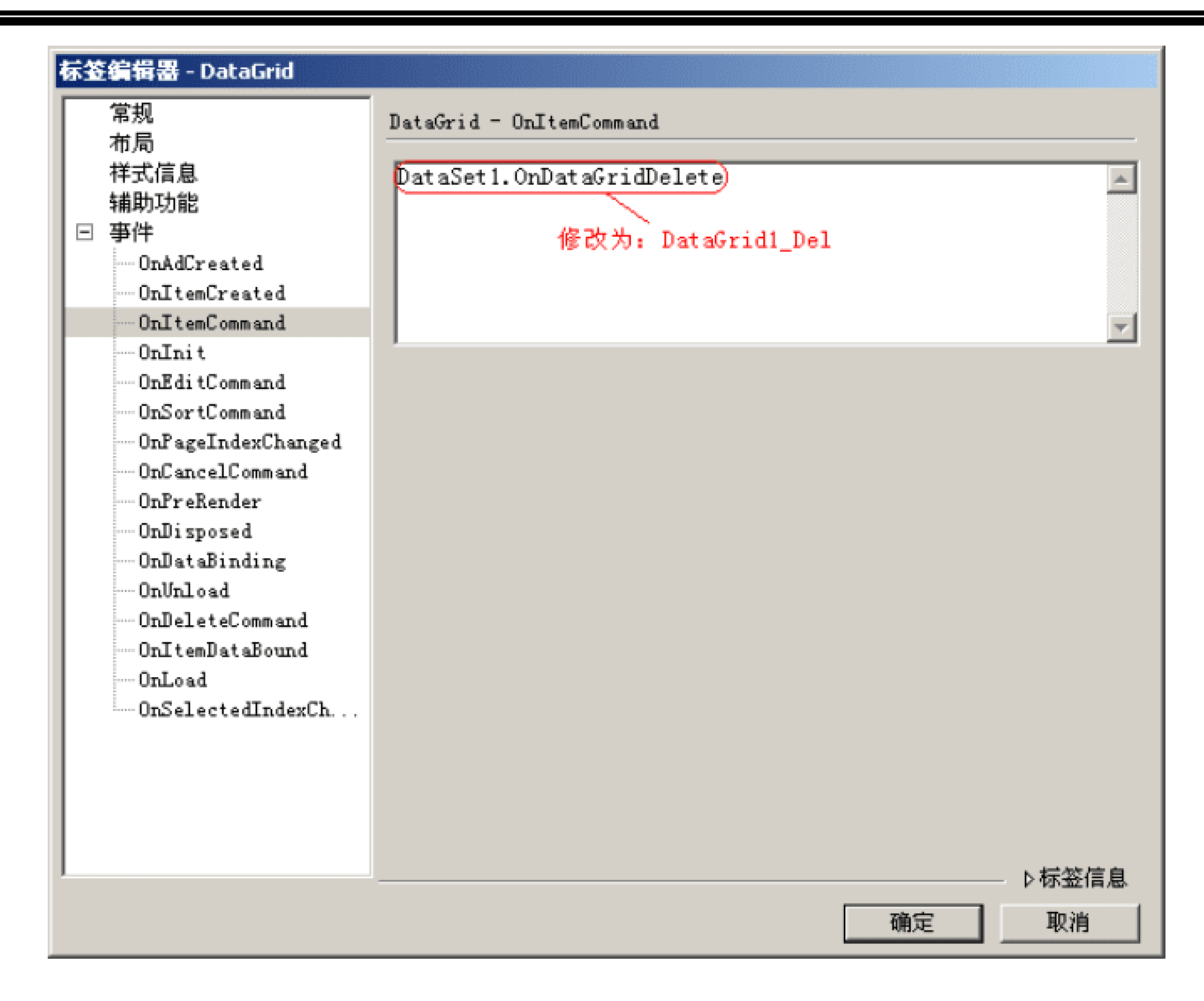


图 11.70　修改 OnItemCommnad 事件

(9)　在【布局】项中设置数据网格的宽度为 90%。单击【确定】按钮，完成数据网格的设置。

提示： 这里修改了数据网格默认的 OnItemCommand 事件。当再次双击该数据网格进行修改时，其 OnItemCommand 事件又将恢复原来的默认值 DataSet1.OnDataGridDelete。这是读者在操作时需要特别注意的。

(10) 切换至【代码】视图，添加自定义的 DataGrid1_Del 事件，其代码如下：

【示例代码】

```
Sub DataGrid1_Del(ByVal Sender As Object, ByVal E As
DataGridCommandEventArgs)
  '获取当前所要删除的新闻分类 ID
  Dim codestr As String = E.Item.Cells(0).Text
  Dim DataR As OleDbDataReader
  Dim Cnn As OleDbConnection
  Dim Cmd As OleDbCommand
  Dim StrCnn As String
  Dim Sql As String
  If CType(e.CommandSource, LinkButton).CommandName = "Delete" Then
    '获取数据库连接字符串
    StrCnn = System.Configuration.ConfigurationSettings.AppSettings
("MM_CONNECTION_STRING_Cnn")
    Cnn = New OleDbConnection(StrCnn)
    Cnn.Open() '打开数据库连接
    '查询是否存在与所要删除的分类相关联的新闻信息
    Sql = "select * from Artinfo where classid=" & Trim(codestr)
    Cmd = New OleDbCommand(Sql, Cnn)
    DataR = Cmd.ExecuteReader()
    If DataR.Read() Then
      '存在与该分类相关的新闻信息，提示无法删除
```

```
        ClientScript.RegisterStartupScript(Me.GetType(), "", GetInfo("存在与
    该分类相关的新闻信息，无法删除！"))
        Cnn.Close()
        Exit Sub
      End if
      DataR.Close()  '关闭 DataReader 对象
      '查询所要删除的新闻分类是否存在子分类信息
      Sql = "select * from classinfo where upclass=" & Trim(codestr)
      Cmd = New OleDbCommand(Sql, Cnn)
      DataR = Cmd.ExecuteReader()
      If DataR.Read() Then
        '该分类存在子分类信息，提示无法删除
        ClientScript.RegisterStartupScript(Me.GetType(), "", GetInfo("该分类
    存在下级分类信息，无法删除！"))
        Cnn.Close()
        Exit Sub
      End if
      DataR.Close()  '关闭 DataReader 对象
      '符合删除条件，执行删除操作
      Sql = "delete from classinfo where classid=" & codestr
      Cmd = New OleDbCommand(Sql, Cnn)
      Cmd.ExecuteNonQuery()
      Cnn.Close()  '关闭数据库连接
      '将页面跳转至本页面，刷新数据的显示
      Response.Redirect("Class_Admin.aspx")
    End If
  End Sub
```

至此，分类管理的页面功能设计完成，页面预览效果如图 11.71 所示。

图 11.71　页面预览效果

也许读者会问，为什么数据网格中的上级分类列只能显示分类 ID，而不能显示分类名称？因为这会让用户感觉很别扭。确实，在正常情况下，上级分类应该显示分类名称的。

但 Dreamweaver 8 中的数据库行为对复杂的 SQL 语句缺乏强有力的支持。事实上，在数据集 DataSet1 中的 SQL 语句添加一个 Select 子查询即可返回上级分类对应的分类名称，而非分类 ID。但在 Dreamweaver 8 的数据网格的可视化操作中，是不支持这条包含子查询的 SQL 语句。当使用该语句作为数据集 DataSet1 的 SQL 语句时，在创建数据网格并选择数据集 DataSet1 时将会提示错误，如图 11.72 所示。

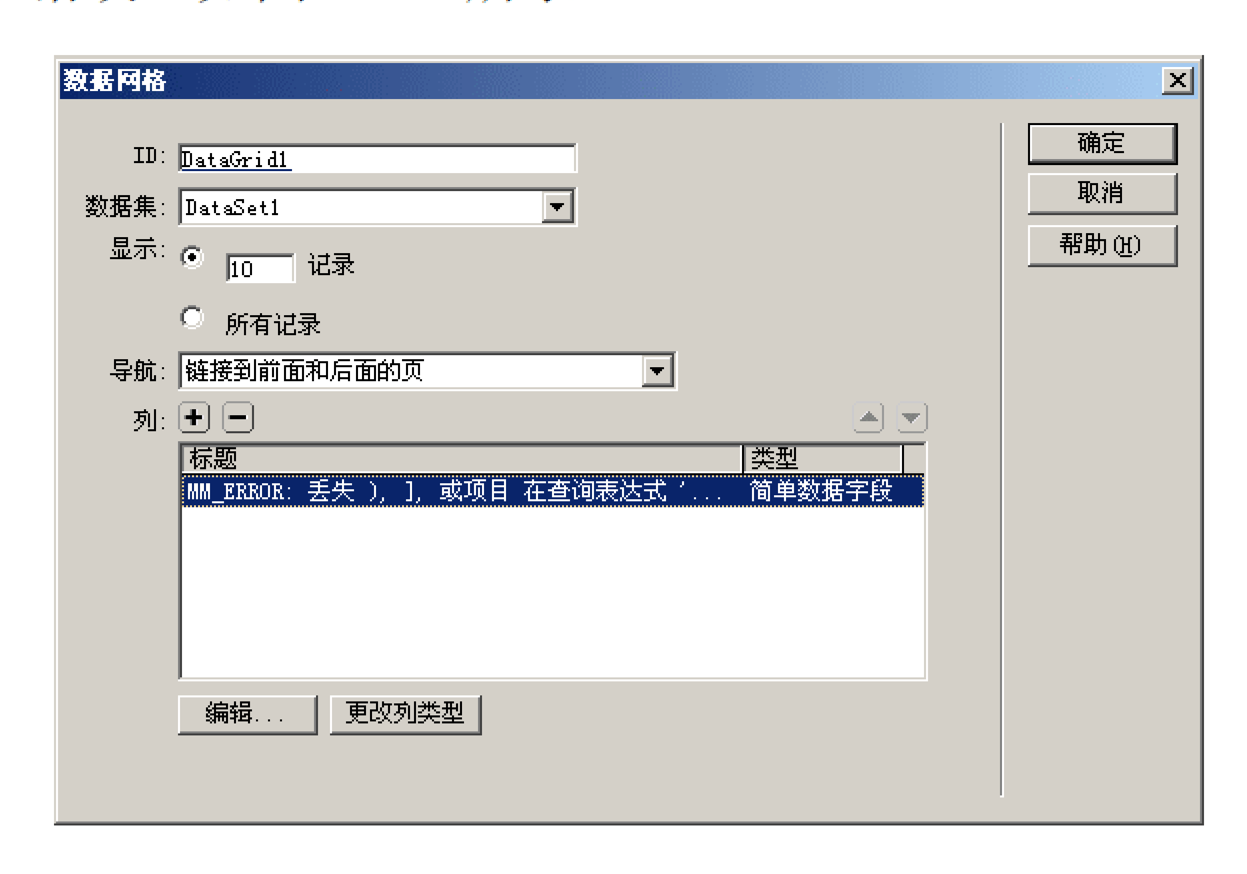


图 11.72　错误提示

这是非常遗憾的。但是，这个问题是可以巧妙解决的。既然在创建数据网格时，无法识别带有复杂 SQL 语句的数据集字段，那么，可以在创建数据网格之后来重新定义数据集的 SQL 语句。前面用户已经实现了数据网格的创建和数据绑定，此时在【服务器行为】面板中双击数据集 DataSet1，在弹出的【数据集】对话框中将其 SQL 语句修改为以下语句：

```
SELECT classid,(select classname from classinfo where classid=a.upclass) as
upclass,classname FROM ClassInfo a ORDER BY UpClass ASC
```

此时，页面预览中将显示上级分类所对应的分类名称，而非分类 ID，如图 11.73 所示。

图 11.73　页面预览

虽然这样已经解决了问题，但是需要注意的是，用户将无法再次通过双击【服务器行

为】面板中的数据网格来对其进行编辑，因为在可视化操作中它依然无法支持带有复杂 SQL 语句的数据集。

11.6.2　添加分类

(1) 新建一个 ASP.NET VB 类型的页面，将其命名为 Class_Add.aspx。选择【修改】|【模板】|【套用模板到页】命令，在弹出的【选择模板】对话框中，选择【后台模板】，将模板应用到本页面。

(2) 在表格 table2 的第 3 个单元格中，插入一个 3 行 2 列的表格 table3，其属性设置如图 11.74 所示。

图 11.74　表格属性

(3) 在表格 table3 的各个单元格中输入相应的文本，并添加用于信息输入的各个 Web 服务器控件，如图 11.75 所示。

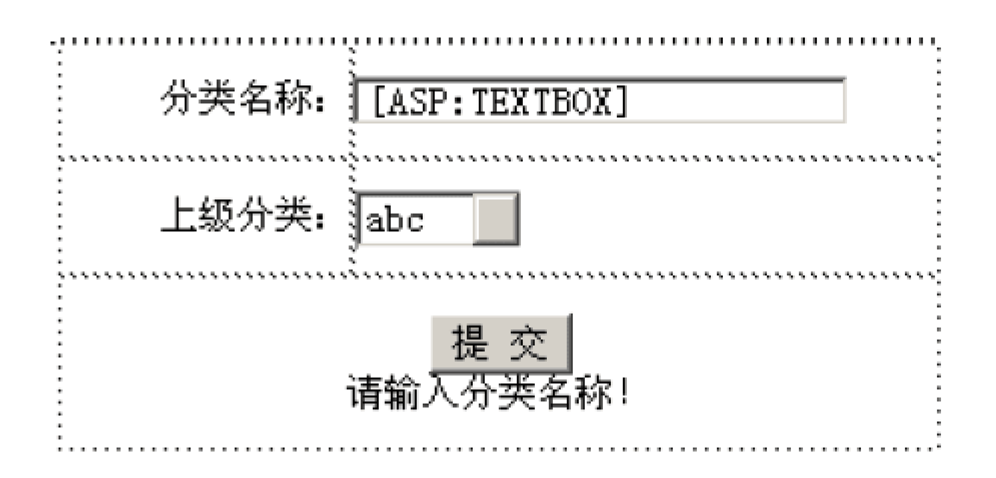


图 11.75　表格设计

(4) 在【提交】按钮之后，添加了一个 RequiredFieldValidator 服务器控件，用于对分类名称的输入进行验证，其属性设置如图 11.76 所示。

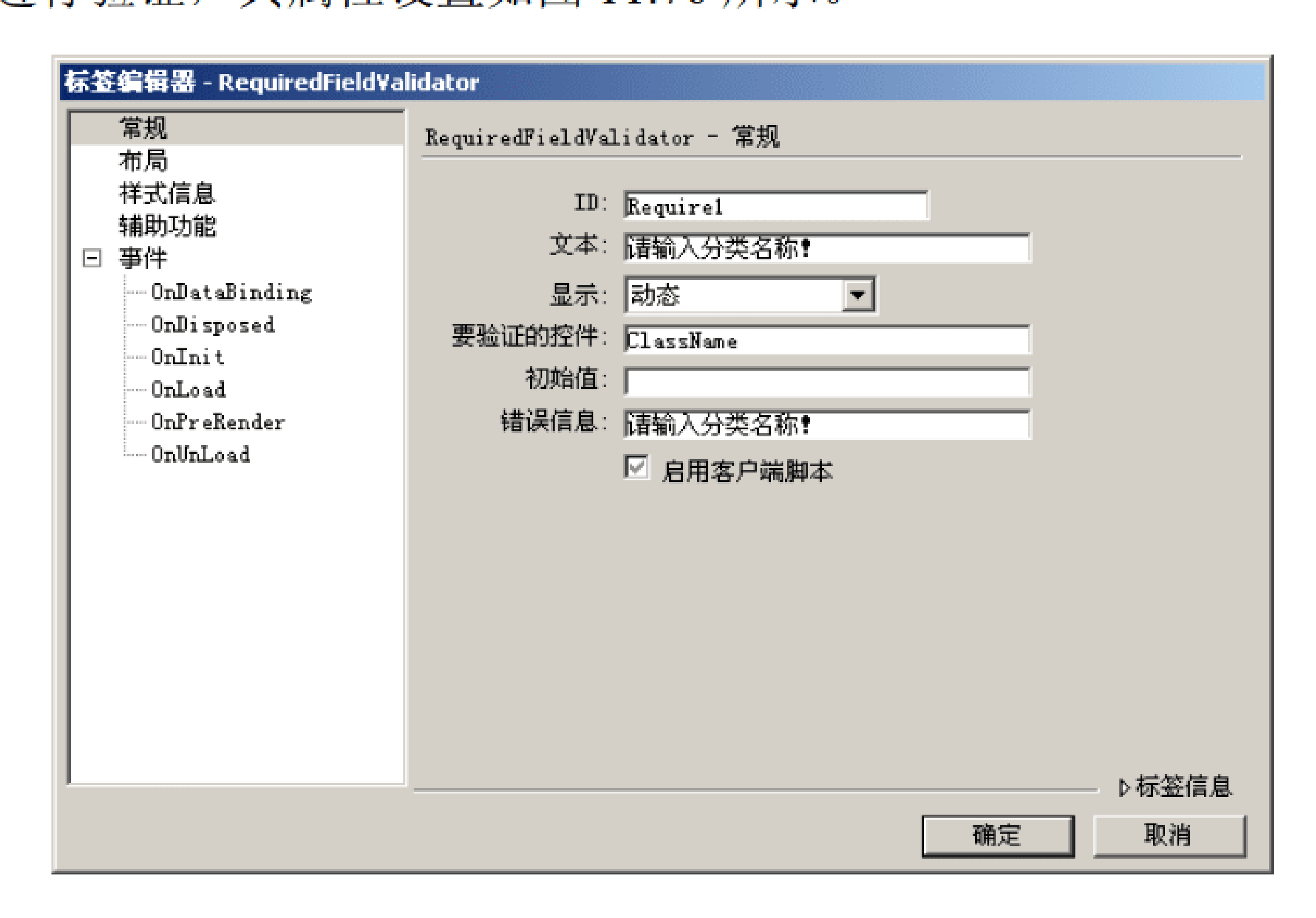


图 11.76　Require1 验证控件

(5) 对于上级分类对应的下拉列表框控件的数据绑定，是在代码中通过自定义过程LoadClass1()来实现的。该过程代码在新闻搜索中已经进行了描述，这里不再赘述。但需要注意的是，如果用户所添加的分类并非其分类的子分类时，此时是无须选择上级分类的。签于这种考虑，需要在上级分类中添加一项“<无>”，以便用户在添加根分类信息时进行选择。为此，需要修改页面的 Page_Load 事件，修改后的代码如下：

【示例代码】

```
Sub Page_Load(Sender As Object,E As EventArgs)
  If Not IsPostBack Then
    '定义 ListItem 变量
    Dim theItem As New ListItem()
    theItem.Text = "<无>"  '设置数据项的显示文本
    theItem.Value = 0  '设置数据项的返回值
    xwfl.Items.Add(theItem)  '将以上定义的数据项添加至上级分类下拉列表框中
    '加载已有的新闻分类信息
    LoadClass1(0,0)
  End If
End Sub
```

在 Page_Load 事件中，将首先添加一项“<无>”至上级分类下拉列表框中，然后再通过自定义过程 LoadClass()来加载已有的新闻分类信息。

(6) 打开【服务器行为】面板，单击 按钮，在弹出的菜单中选择【插入记录】。此时，将弹出【插入记录】对话框，如图 11.77 所示。

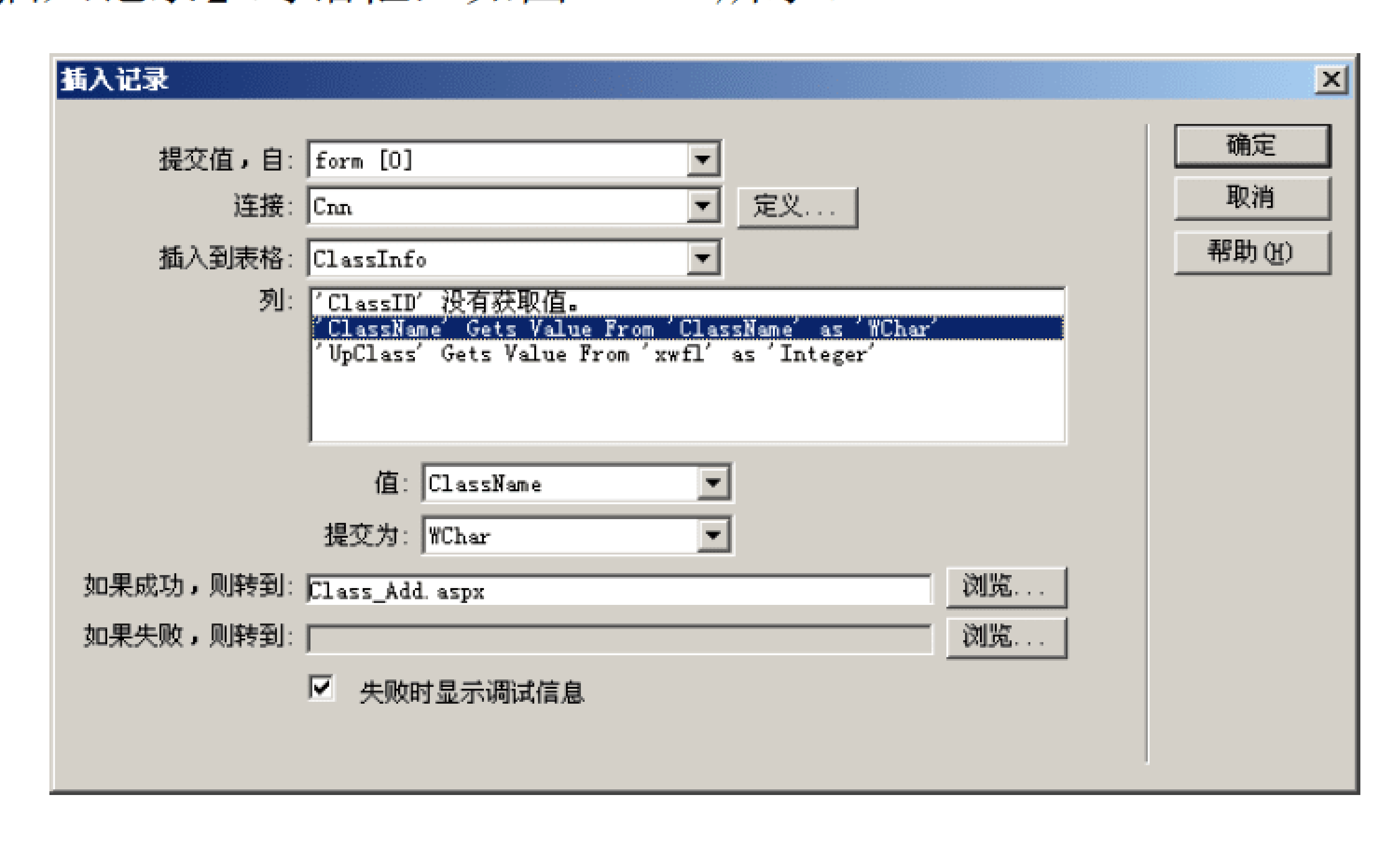


图 11.77　【插入记录】对话框

数据库连接选择 Cnn，选择表格 ClassInfo。然后，分别设置字段 ClassName 和 UpClass 的获取值为表单元素 ClassName 和 xwfl 的值。最后，设置当数据提交成功后所转向的页面为 Class_Add.aspx，即仍返回本页面。

(7) 单击【确定】按钮，完成插入记录的创建。

至此，添加分类页面的功能已实现，其页面预览如图 11.78 所示。

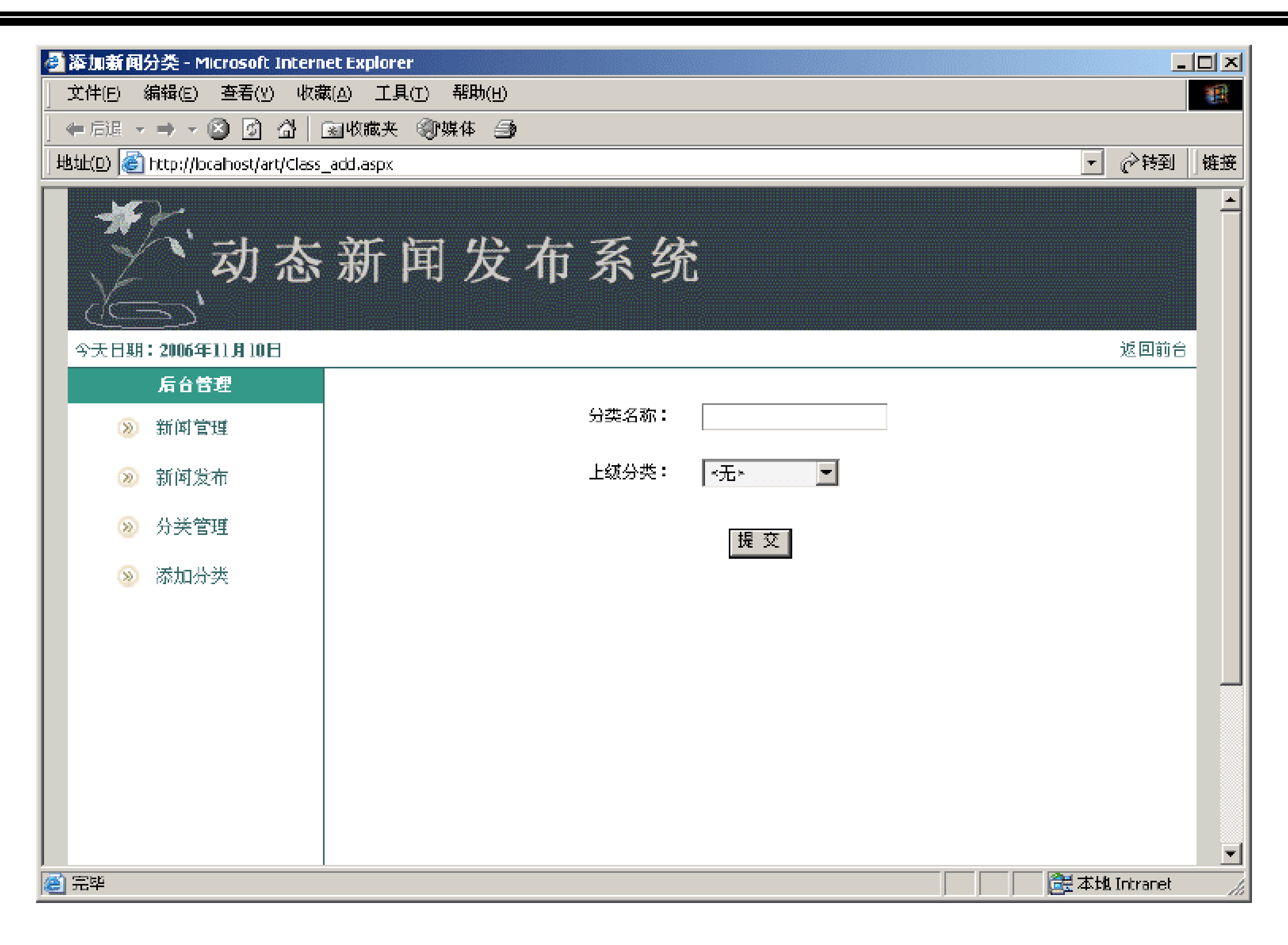


图 11.78　页面预览

11.7　新闻页面编辑

在新闻管理中，曾经提到，对新闻信息的编辑操作是将页面跳转至新闻编辑页面 Art_Edit.aspx 来执行的。本节将讲解新闻编辑页面 Art_Edit.aspx 的具体实现。

新闻编辑页面与新闻发布页面的布局较为相似，因此可先通过复制新闻发布页面来建立新闻编辑页面，然后在原有基础上进行修改。

(1) 打开新闻发布页面 Art_Add.aspx，选择【文件】|【另存为】命令，在弹出的【另存为】对话框中输入文件名 Art_Edit.aspx，并单击【保存】按钮，完成另存为操作。

(2) 在新闻编辑页面中，首先将页面标题设置为“新闻编辑”，然后删除页面中的【插入记录】服务器行为。

(3) 在对指定的新闻进行编辑之前，首先必须获取指定新闻的相关信息，并将其显示在页面的各个输入控件中，以便用户进行修改。为此，需要创建一个数据集。

(4) 打开【服务器行为】面板，单击 + 按钮，从弹出的菜单中选择【数据集】命令。在弹出的【数据集】对话框中设置数据集名称为 DataSet1，选择连接 Cnn，表格设置为 ArtInfo，筛选条件为字段 ID 等于 URL 参数 ID，如图 11.79 所示。

(5) 单击【确定】按钮，完成数据集 DataSet1 的创建，该数据集将返回指定所要编辑的新闻的相关信息。

(6) 在【设计】视图中，选择新闻标题对应的文本框控件并右击，从弹出的快捷菜单中选择【编辑标签】命令，并在弹出的【标签编辑器】对话框中设置其文本为“<%# DataSet1.FieldValue("ArtTitle", Container) %>”，如图 11.80 所示。

(7) 重复以上操作，分别设置“新闻内容”、“新闻作者”、“新闻来源”和“关键字”所对应的文本输入框的值，其对应的动态文本分别为字段 ArtContent、Author、CopyFrom 和 Keys。

图 11.79　创建数据集 DataSet1

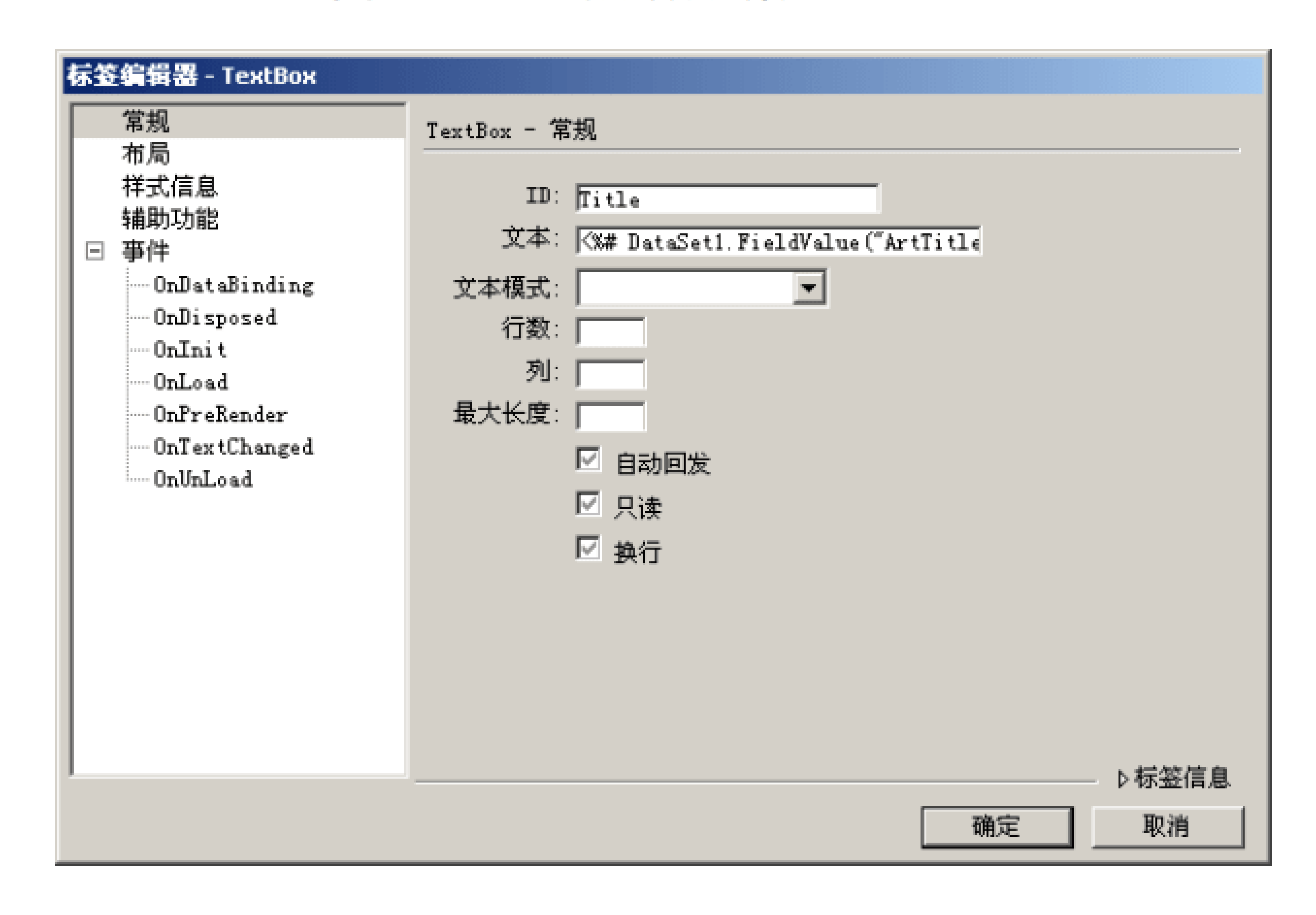


图 11.80　【标签编辑器】对话框

(8) 为了在更新记录时获取当前进行编辑操作的新闻 ID，在【提交】按钮之后插入一个隐藏域，其名称设置为 ArtID，其值设置为“<%# Request.QueryString("id")%>”，如图 11.81 所示。

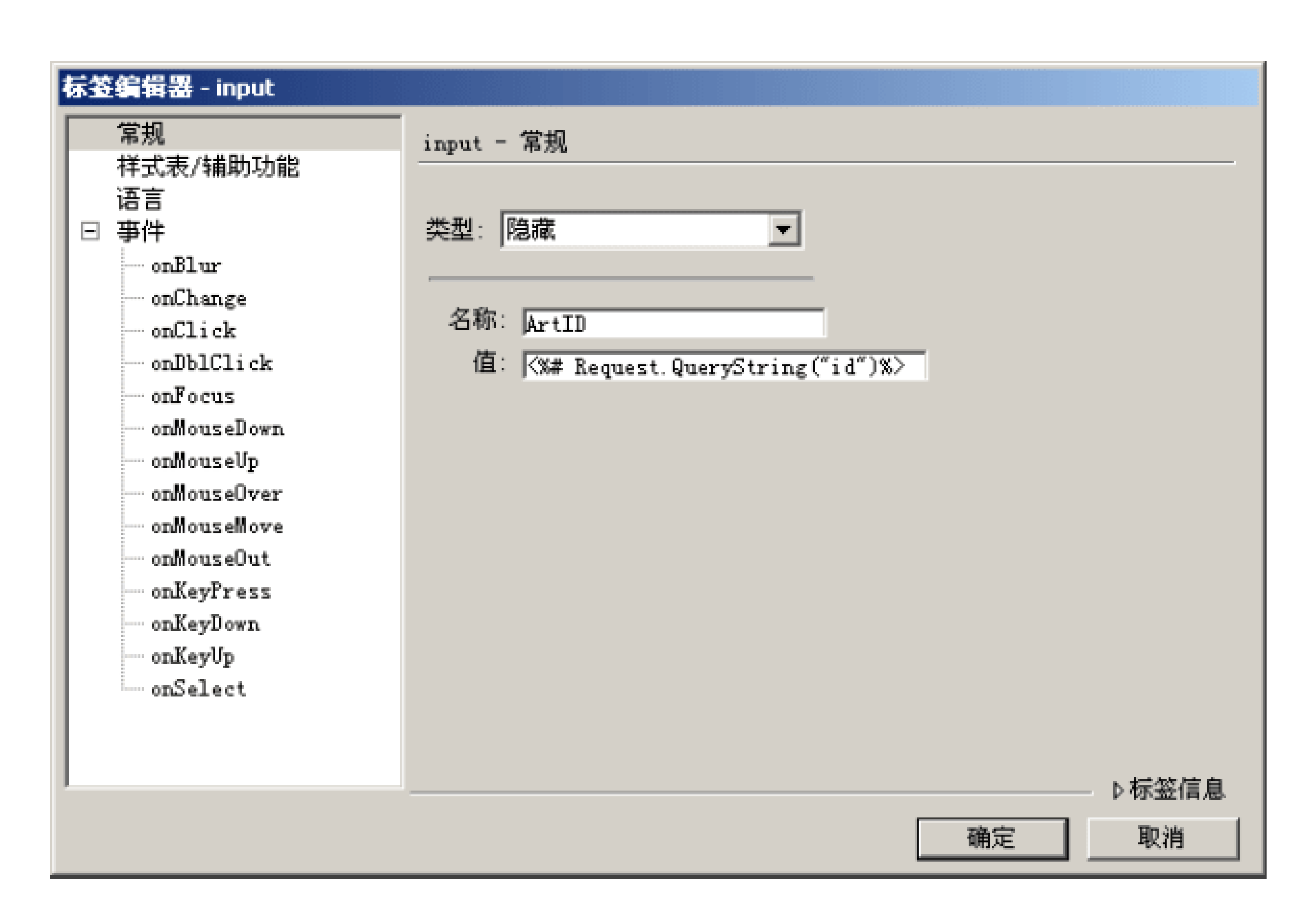


图 11.81　插入隐藏域

(9) 在页面中，新闻分类和通过审核所对应的均为下拉列表框控件。在加载其数据的同时，还需要在其数据中寻找当前编辑的新闻信息所对应的新闻分类和审核标志，并将其置为下拉列表框控件的当前选中项。此操作只能通过代码来实现，为此需要修改页面的 Page_Load 事件，修改后的代码如下：

【示例代码】

```
Sub Page_Load(Sender As Object,E As EventArgs)
  Dim theClass as Integer
  Dim theSH as Integer
  If Not IsPostBack Then
    '加载新闻分类信息
    LoadClass1(0,0)
    '判断数据集 DataSet1 的记录数是否大于 0，即当前所要编辑的新闻是否存在
    If DataSet1.DefaultView.Table.Rows.Count>0 Then
      '获取当前所要编辑的新闻的分类 ID
      theClass=DataSet1.DefaultView.Table.Rows(0)("classid")
      '获取当前所要编辑的新闻的审核标志
      theSH=DataSet1.DefaultView.Table.Rows(0)("sh")
      if not (xwfl.Items.FindByValue(theClass) Is Nothing) then
        '在 xwfl 下拉框控件中找到与 theClass 匹配的数据项，并设置其 Selected 属性为
True
        xwfl.Items.FindByValue(theClass).Selected = True
      end if
      if not (sh.Items.FindByValue(theSH) Is Nothing) then
        '在 SH 下拉框控件中找到与 theSH 匹配的数据项，并设置其 Selected 属性为 True
        sh.Items.FindByValue(theSH).Selected = True
      end if
    End If
  End If
End Sub
```

在绑定新闻信息的数据显示之后，接下来实现数据的更新。

(10) 在【应用程序服务器行为】面板中，单击 + 按钮，从弹出的菜单中选择【更新记录】命令。在弹出的【更新记录】对话框中，设置连接为 Cnn，更新表格为 ArtInfo。分别设置字段 ArtContent、ArtTitle、Author、ClassID、CopyFrom、Keys 和 SH 的获取值为表单元素 Content、Title、Author、xwfl、CopyFrom 和 KeyWord 的值；而作为主键的字段 ID，则设置其获取值为隐藏域 ArtID 的值。最后，设置数据提交成功后转到的页面为新闻管理页面 xwgl.aspx，如图 11.82 所示。

单击【确定】按钮，完成更新记录的创建。

新闻编辑页面的功能已基本完成，但还需考虑当用户所选择的新闻 ID 不存在的情况(比如说，用户没有选择所要编辑的新闻，而是直接输入地址进入本页面)。

(11) 在控制新闻信息显示的表格 table3 之后输入文本“请选择所要编辑的新闻！”，并设置其字体颜色为红色。然后，选择该文本，在【服务器行为】面板中，单击 + 按钮，从弹出的菜单中选择【显示区域】|【数据集为空时显示】命令，并在弹出的对话框中设置数据集为 DataSet1，如图 11.83 所示。

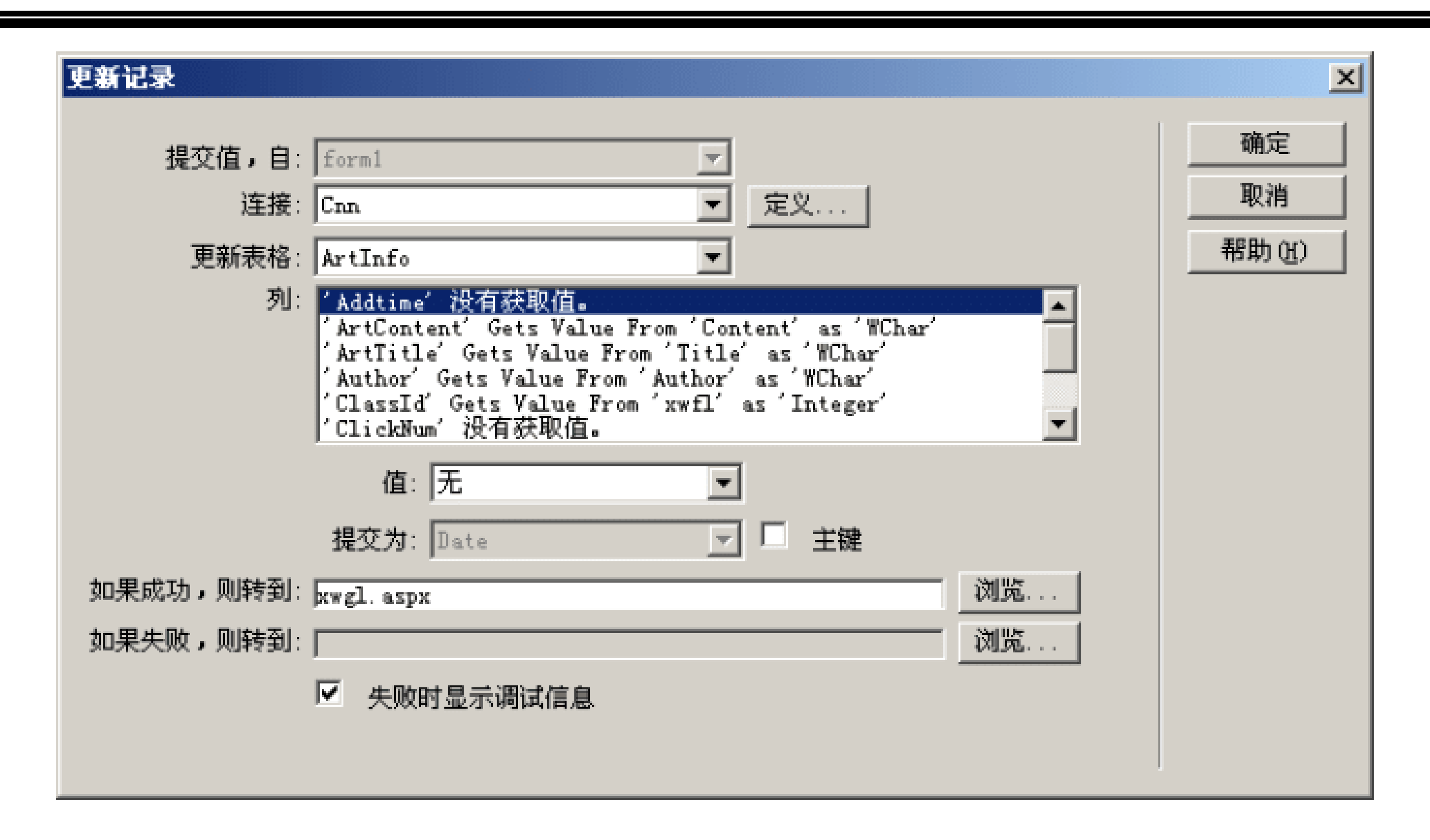


图 11.82　【更新记录】对话框

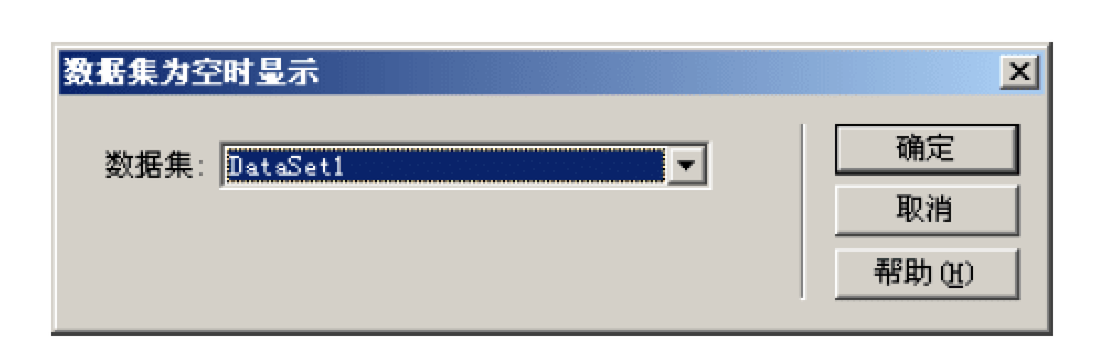


图 11.83　【数据集为空时显示】对话框

单击【确定】按钮，完成显示区域的创建。

至此，新闻编辑页面制作完成，其页面预览如图 11.84 所示。

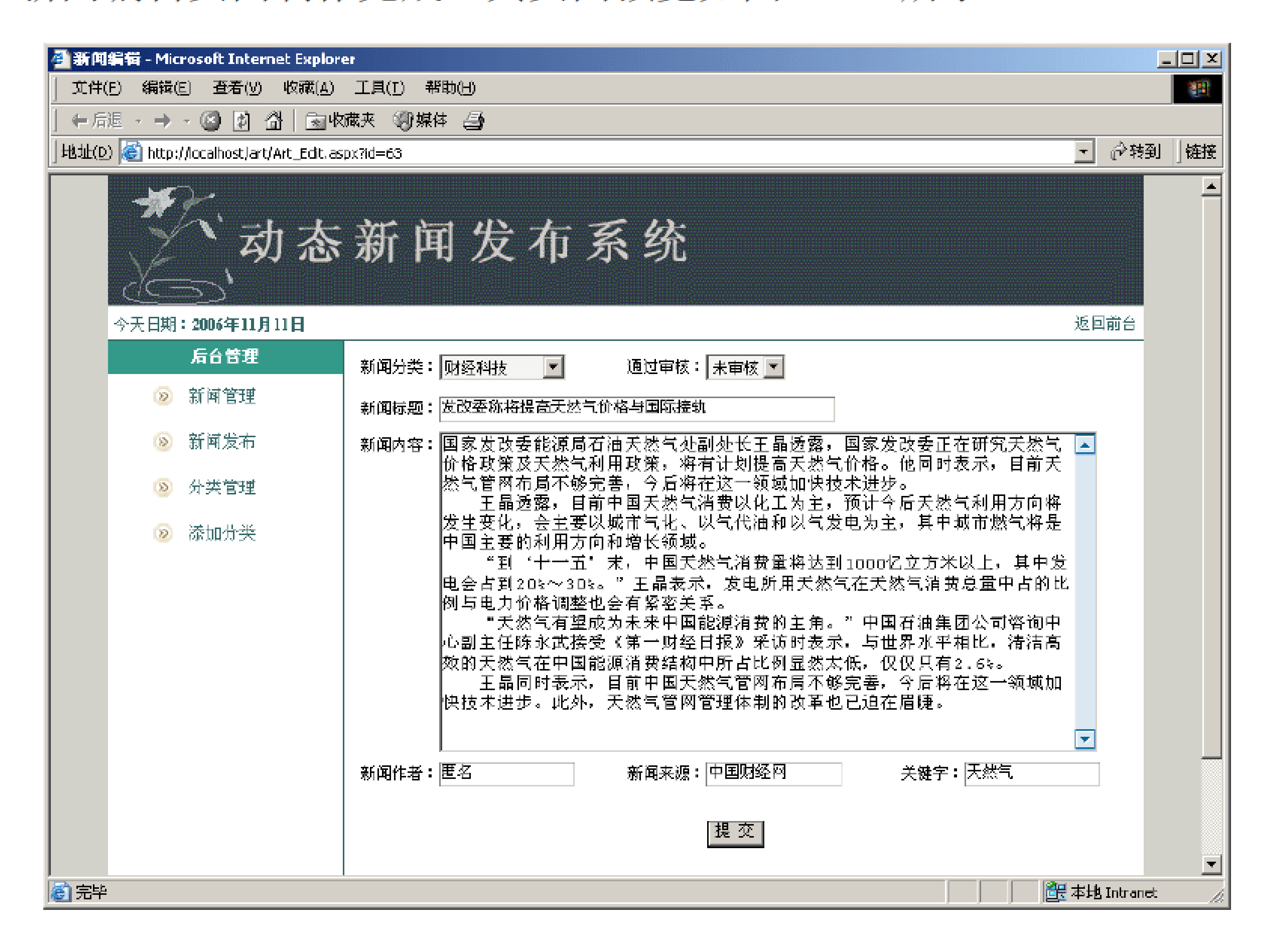


图 11.84　页面预览

11.8　习　　题

(1) 扩展新闻的搜索功能，添加发布时间和新闻来源两个搜索条件。

(2) 添加后台模块的管理员登录页面，并将前台页面的【管理】链接指向该页面。